Norm Awareness for Virtual Characters Behaviour: A Socio-Cognitive Approach

Lee, Jeehang

Award date:
2015

Awarding institution:
University of Bath

Norm Awareness for Virtual Characters Behaviour

- A Socio-Cognitive Approach -

submitted by

JeeHang Lee

for the degree of Doctor of Philosophy

of the

University of Bath

Department of Computer Science

January 2015

SUMMARY

Social intelligence has a huge impact on the determination of human behaviour in the society. The use of norms can contribute to advances in this social intelligence by the provision of appropriate behaviour based upon the understanding of social situations. Hence, the domain of virtual characters research has given much attention to take advantage of these characteristics of norms particularly in engineering human-like behaviour. However, a lack of capability in reasoning about norms as well as a lack of norm autonomy in virtual characters have significantly diminished the naturalism in virtual characters behaviour. Within this context, a hybrid approach incorporating social and individual reasoning inspired by socio-cognitive theory is taken into account in this thesis. To this end, we propose **DNA**^{3}, Distributed Norm Aware Agent Architecture, established through the integration of (i) the institution, a normative framework performing the social reasoning, (ii) ***N-Jason***, a (BDI-type) cognitive agent carrying out run-time norm-aware deliberation and (iii) a virtual character in charge of perception and realisation of actions.

The institution takes responsibility of (i) analysis of state of external worlds by recording a sequence of event occurrences observed by multiple virtual agents, (ii) reasoning about situationally appropriate behaviour with an assistance from Answer Set Programming (ASP) solver depending upon the social context virtual characters encounter and (iii) in turn detachment of a new set of norms, more precisely normative consequences of specific actions, to virtual characters. This contributes to the enhancement in the flexibility in specifying and reasoning about social norms subject to changes of social situations. Those detached norms are involved in the reasoning process of individual virtual characters. In here, a norm-aware BDI-type agent, ***N-Jason***, performs a practical reasoning to select a plan to execute between norms and goals. Basically, ***N-Jason*** offers a generic norm execution mechanism on top of norm aware deliberation to contribute to the exploitation of run-time norm compliance. The selection of agent

behaviour is achieved in the norm-aware deliberation process by *intention scheduling* with deadlines and priorities. This improves the rationality in the choice of behaviour with taking into account the preference on norms and goals in agent mind by evaluation of the importance and imminence between feasible plans triggered by both norms and goals.

The design and simulation of politeness is presented as an evaluation of **DNA**^{3} with respect to the effectiveness and adequacy in modelling virtual characters behaviour. The emphasis in here lies on the capability that is able to exhibit different types of appropriate polite behaviour in response to frequent changes in social situations. This is mainly driven by two main activities: prediction of other participants' intention is carried out by norm-aware virtual characters whilst the understanding of context and reasoning about relevant social behaviour is performed in normative frameworks. For this purpose, three case studies are provided in this thesis: (i) politeness in navigation of individuals, (ii) politeness in the formation and navigation of groups during a guided tour, and(iii) evacuation model as a politeness in the emergency situation. The evaluation is conducted by measuring: (i) the appropriateness of in response to scenarios (e.g. a number of avoiding collisions) and (ii) the reliability of agent decision making (e.g. a response time in relation to norms with the highest priority and the most urgent).

ACKNOWLEDGEMENT

First and foremost, my sincere appreciation goes to my supervisor, Dr. Julian Padget, for his constant support and encouragement. He is always an enthusiastic supporter for my innovative ideas, extremely encouraging for me to overcome obstacles, and full of wisdom for the guidance to reach the destination. His helpful comments and suggestions certainly were a valuable source for this thesis. I also would like to thank him for the help in establishing research collaborations and international contacts. My special thanks go to Dr. Joanna Bryson, my co-supervisor, for her constant support and help throughout the year. Further, I want to thank Prof. Dr. Eamonn O'Neill for his kind and careful supervision during Julian's sabbatical away.

I am very thankful to Dr. Brian Logan, Dr. Natasha Alechina and Daniela Dybalova, for the possibility to conduct the research at the University of Nottingham, which gave me the opportunity to learn the deep understanding of agents. I also want to thank Dr. Alicia HyunSun Kim for the support and advices with full of wisdom on this journey. Additionally, I thank Dr. Alessio Guglielmi for giving me the lecturing opportunity which completes the other side of my ph.d.

I would like to express my thankfulness to all colleagues, Dr. Tingting Li, Dr. Saeid Ardakani, Dr. Esraa Alwan, Dr. Qi Wu, Jekaterina Novikova, Dr. Garry Ren and Dr. Bidan Huang, who were always my side to keep my ph.d. bright and endurable. I specially thank Dr. Gideon Bibu, Dr. Han Gong, Dr. GeonHwan Cho, Dr. SungHa Hong, Dr. MinSu Cho and SunHyik Ahn for their constant support and prayer.

Above all, I thank my parents for their encouragement, understanding, support and love which have helped me throughout my life. Last but not least, my deepest gratitude goes to the family, my dearest YuJin Lim and my gorgeous son KangOn Lee. Without their life support, endless patient, encouragement, belief and unconditional love, this thesis would not have become a reality. Yes, God, you have done it all. Full of glory for this is all yours.

Contents

List of Figures

List of Tables

List of Algorithms

Listings

Chapter 1

Introduction

Social intelligence is an essential ingredient in human intelligence [Bandura, 2005, Dautenhahn, 2007]. It plays a substantial role in decision making which influences thought and action in the various social situations people encounter. Norms have a potential to contribute to advances in social intelligence of agents since not only do norms offer guidance about correct behaviour in social situations, they also provide better understanding about a current social interaction. Hence, it might be seen that the addition of normative reasoning to virtual characters can be regarded as a way in which to improve characters' decision making and thus to perform enhanced responses which appear more socially acceptable.

This thesis aims to design and simulate human-like behaviour in virtual characters taking into account the benefits arising from the use of norms. We basically pursue a socio-cognitive approach incorporating a social and a cognitive dimension into the reasoning process of virtual characters behaviour to this end [Stein, 1993, Bandura, 2001, Akgün et al., 2003]. Therefore, we describe a computational model, Distributed Norm-Aware Agent Architecture (**DNA**^{3}), established by norm-aware (BDI-type) cognitive agents under the governance of normative frameworks for the purpose of engineering norm-aware Intelligent Virtual Agents (IVAs). Given **DNA**^{3}, we intend IVA behaviour to mimic better – in the way socio-cognitive theory addresses it – how humans behave under various social settings, through the combination of each IVA's own contextual information, constructed by its cognition (which represents the cognitive dimension) and socially constructed beliefs (e.g. norms or expectation) from other participants (which represents the social dimension).

In this chapter, we firstly present the motivation for the research in Section 1.1, then the problems we would like to solve are described in Section 1.2. It is followed by the contribution of the approaches in this thesis in Section 1.3. After the approach in Section 1.4, the whole structure of this thesis is given in Section 1.5. At the end of the chapter, related publications which are the basis of this thesis are discussed in Section 1.6.

1.1 Motivation

Virtual environments (VEs) have seen significant developments in recent years, moving from virtual worlds designed for specific purposes (e.g. on-line games) to more general-purpose environments which are not for entertainment but for serious games, resulting in changes in form, scope and purpose [Messinger et al., 2009]. With these changes in application domain has come a demand for more human-like virtual characters in the context of more sophisticated and dynamic VEs.

The human-likeness stands for a conception that virtual worlds and its participants should bear a close resemblance to human behaviours in the real world. Consequently, two approaches are widely accepted as a prerequisite: (i) to pursue realism in appearance through computer graphics research, on the one hand, and (ii) to build naturalism in behaviours as an outcome of AI on the other. Since the former, realism in appearance, has been more or less satisfied due to the dramatic development of graphics technology, the latter, naturalism in virtual characters behaviour, is lately being paid more attention for the enhancement of human-likeness.

In AI domains, the embodiment of human-like behaviour has been a long term challenge. One main theme to this end is to simulate and/or replicate the social aspect of human intelligence [Payr and Trappl, 2004], which has been given more attention in order to make robots/agents appear smarter (i.e. looks more human-like or believable in behaviour). In so doing, the robots/agents can exhibit improved responses (i.e. socially more acceptable behaviour) to the other participants (e.g. humans, robots, agents, intelligent softwares) in that circumstance [Walters et al., 2008].

The use of norms has an impact on improving the social intelligence of robots and (software) agents. In principle, norms are socially constructed beliefs which represent patterns of appropriate actions in various social settings. Norms can influence human

thought and action by means of the deontic forces they impose, which indicate '*the way things are* or *the way things should be*' [Stein, 1997]. In addition, norms could be described as human "*expectations*" during social interactions. In other words, norms are able to act as a window to interpret the associated (tangible) contents (e.g. the information and transactions participants communicate each other) of an interaction [Mitchell, 1973]. Therefore, to consider those characteristics of norms ensures the appropriateness in behaviour with respect to the mutual expectation which manifest a proper choice on either a desire individuals want to pursue or an action they can actually do in the social context. [Stein, 1993].

One interesting finding in virtual environment domains is that norms can influence virtual character behaviour as humans do in the real human society. As Yee *et al.* [Yee et al., 2007] show, social behaviours of human controlled characters with regards to Inter-Personal Distance (IPD) are governed by the same social norms as those in the real human society. Depending on gender, intimacy or cultural background, player characters (PCs) exhibit different levels of IPD, such as whether or not to keep a certain distance, to avoid eye contacts between avatars or to change the avatar's focus of attention in response to the inter-personal context, which shows a very similar behaviour pattern as those that humans perform in the real world. Friedman *et al.* [Friedman et al., 2007] also ensure this aspect in virtual characters behaviour with the investigation of the spatial distance between human controlled avatars during the conversation with others.

Whilst those observations are rather focused on inter-relational norms, there is another example that norms also have an affect on conventional (social) behaviour. Samperi *et al.* [Samperi et al., 2012] discover the particular patterns in navigation of PCs. Interestingly, PCs perform exactly the same behaviour as we can see in the real world when people explore the parks or gardens. They are likely to keep to the path or road rather than wandering on the grass (or green) area while they are navigating to a destination, despite the shortest path obviously running across the grassed area. Although there might not be sanctions or punishment in the case of the violation of norms, players seem likely to control their avatar to do the "right thing" as players would normally in a park.

This implies that if we wish the behaviour of virtual characters (e.g. non-player characters (NPCs)) to resemble human behaviour more closely, then it is desirable

somehow to convey those real world human social norms into the agent mind. Hence, we believe that to take into account an impact (or deontic forces) of norm on human behaviour promises better human-likeness in the design and simulation of NPCs behaviour.

1.2 Problem Statement

In the domain of virtual environments, the use of norms for NPCs behaviour is commonly achieved by two approaches:

Approach (1) The one is to make use of individual agent reasoning independently. In this case, NPCs have in their mind pre-defined, situation specific norms and their associated plans, in order to be able to exhibit norm compliant behaviour. When NPCs perceive actions and events occuring in the virtual world, NPCs carry out reactive decision making whether to comply norms or not subject to the current percept [Pedica and Högni Vilhjélmsson, 2010, Si et al., 2006].

Approach (2) The other is to make use of social (norm) reasoning via normative frameworks (also called institutions) [Bogdanovych et al., 2008b, Almajano et al., 2013], or in conjunction with individual agent reasoning [Ranathunga, 2013, Baines and Padget, 2014]. The normative frameworks are able to perform automated norm reasoning in response to observations from the external environment (virtual worlds in this case), determine which norms are the most appropriate in the current situation and detach those norms into NPCs' mind. Whenever new norms are detached, the receiving NPC perform its individual reasoning.

In general, individual reasoning which actually refers to decision making in NPCs mind brings about an associated behaviour in an open environment. According to Russel and Norvig [Russell and Norvig, 2002], the decision making is usually performed by three types of agents: (i) Reactive agents, (ii) Utility based agents and (iii) Goal oriented agents. In the domain of NPCs research, NPC behaviour is also typically determined by those agent models. For example, the literature contains approaches to design reactive plans associated with norms via reactive agents [Pedica and

Högni Vilhjélmsson, 2010], to program goals and plans leading to norm-compliant behaviour [Ranathunga, 2013, Baines and Padget, 2014] with goal oriented agents, or to provide probatility and utility distribution of actions for planning to norm compliant behaviour [Si et al., 2006] based on utility oriented agents. Although such approaches have become a standard means to design AI with norms in NPC behaviour, it gives rise to a couple of issues in decision making with norms.

On the one hand, in both cases (approach 1 and 2), NPCs are likely to be regimented by design. In other words, the most significant point, in our view, is that agents do not have the possibility to reason about norm compliance: they are required simply to follow pre-defined behaviours to comply with norms without deliberation on norms, goals and potential sanctions. This may cause deprivation of agent (norm) autonomy since pre-defined behaviours in effect are treated as hard constraints, whose violation is not possible.

On the other hand, particularly in approach 1, norms in this type of agents are too domain specific to be adaptive subject to the changes of situations. In other words, norms are usually a pre-defined set of desirable behaviour dedicated to the specific situations in the virtual environments. When the situation is changed, the norms in the agent mind might be no longer valid in the new situation subsequently. To be able to behave correctly '*in real time*', '*at right places*' and '*being associated with interactions amongst other participants*' in response to the changes of situation, reasoning about norms themselves should be considered.

Within this context, we have four questions in mind with regards to the above two issues:

(Q1) How to reason about norms subject to dynamic changes of situations in which virtual characters are involved,

(Q2) How to share the new set of norms (constructed by collectives) with virtual characters at run-time,

(Q3) How to ensure norm-aware decision making in virtual characters behaviour, and

(Q4) How to share virtual characters' percepts with both individual agents reasoning and social norm reasoning at run-time.

1.3 Contributions

The central mission of the thesis is to design and simulate norm-aware behaviour of virtual characters. We believe that the consideration of norms and the values they impose ensures more socially acceptable responses from virtual character, so that it enhances the behaviour of virtual characters to make it more human-like with respect to social itelligence.

As mentioned earlier in Section 1.2, the involvement of norms in virtual characters' decision making, according to published work, can be accomplished by either social (norms) reasoning or individual agents reasoning. We pursue a hybrid approach inspired by socio-cognitive theory [Stein, 1993] (or social cognitive theory [Bandura, 2001]), a combination of the two, in this thesis. We begin to set out the motivation and justification in section 1.4, below.

As a result, the primary contributions of the thesis is:

> "The provision of a Distributed Norm-Aware Agent Architecture (**DNA**^{3}) via the combination of cognitive and social reasoning in order to engineer norm-aware behaviour in virtual characters."

DNA^3 facilitates a socio-cognitive perspective on agent decision-making by means of the combination of social norms reasoning and individual cognitive reasoning. In practice, this is a programming framework for norm-aware virtual characters behaviour which consists of:

1. institutions, which are normative frameworks that perform social reasoning,
2. ***N-Jason***, which is a BDI-type cognitive agent carrying out the norm-aware individual reasoning and
3. virtual characters that are in charge of perception and realisation of actions in virtual environments.

Institutions (also called normative frameworks) can be viewed as a kind of external repository of (normative) knowledge from which guidance may be delivered to agents. It is composed of a set of rules whose purpose is the governance of individual agents in the society. These rules are not just hard-coded recipes presenting reactive

behaviours (such as those in the static expert systems), but which rather describe consequences in response to knowledge or observations for reasoning about the current context, resulting in situation-specific norms. The framework not only identifies correct and incorrect actions but also normative facts such as obligations, permissions and prohibitions through the institutional trace, that records the institutions evolving internal state, subject to observed external events captured from the external world. The main contribution of institutions given those characteristics is as follows:

C1-1 To design and specify norms and external events triggerings these norms as perceived by NPCs in the virtual environment,

C1-2 To perform automated norm reasoning, i.e. which norms are the most appropriate in the situation NPCs and humans are encountering and

C1-3 To share social information by means of detachment of (social) norms to NPCs.

The virtual character's behaviour is planned and executed through decision making in (BDI-type) cognitive agents, when institutions bring about new norms (more precisely normative consequences of specific actions) via a social reasoning technique with the assistance of an Answer Set Programming solver. Consequently, such detached norms are involved in the practical reasoning process of individual agents.

In this system, a norm-aware BDI-type agent, ***N-Jason***, performs practical reasoning to select a plan to execute incorporating norms and goals. Basically, ***N-Jason*** offers a run-time norm execution mechanism on top of norm aware deliberation to contribute to the exploitation of run-time norm compliance. Thus, it is capable of the operationalization of new and unknown (event-based) norms not stated in the agent program at run-time by judging the executability of those norms. At the same time, the selection of agent behaviour is achieved in the norm-aware deliberation process by *intention scheduling* with deadlines, priorities and prohibitions. It enables the evaluation of the importance and imminence between feasible intentions triggered by both detached norms and goals, which confirms the decision about which behaviour an agent would prefer between goals, norms and sanctions. In conclusion, the main contribution of ***N-Jason*** is as follows:

C2-1 To provide run-time norm execution on top of the BDI practical reasoning and

C2-2 To carry out norm-aware deliberation (on norms, goals and sanctions).

Lastly, one another (practical) contribution of this thesis is the proposal of a middleware, the Bath-Sensor-Framework (BSF). This BSF facilitates the integration of heterogeneous software systems. Heterogeneity stands for the differences in programming languages to develop the software and platforms (or operating systems) to run software in this context. Given both pub/sub and message based communication on top of standard network protocol (XMPP), BSF enables the integration of cognitive agents, virtual agents and normative frameworks by supporting multiple programming lanuguages (e.g. Java, C# and JavaScript) as well as distributed physical locations. In summary, the contribution of BSF is:

C3-1 Middleware and engineering methodology to integrate normative frameworks, norm-aware agents, NPCs and virtual environments.

C3-2 To establish the communication channels enabling run-time sharing mechanism for both observations and social information.

1.4 Approach

As introduced in the previous section (1.3), we propose a hybrid approach to the engineering of norm-aware virtual characters. This hybrid approach is inspired by socio-cognitive theory [Stein, 1993, Bandura, 2001], which basically emphasises the incorporation of a social and a cognitive dimensions to human choices in society.

Without doubt, "*People do not live their life only in individual autonomy*" [Bandura, 2001]. People tend to give up their autonomy (i.e. intention to achieve individual goals) when they reckon that a certain action which should be done is more socially appropriate (or fits better with moral purposes) than those triggered by their individual desire (i.e. actions that they really want to do). It reminds us that human choice and subsequent behaviour are definitely affected by social influences by an individual's interpretation of social reality [Stein, 1993] during the individual reasoning processes.

The socio-cognitive theory addresses this aspect that the human choice in practice is an outcome of the interplay between individual knowledge and social structures [Cook and Yanow, 1993, Easterby-Smith, 1997]. In other words, it is essential in the deter-

mination of human behaviour to consider not only individual collectives (e.g. cognitive structure) constructed by cognitive process performed in human brain [Anderson, 2005] but also socially constructed belief (e.g. social norms) established by social process taking place in institutions [Stein, 1997] or organisations [Akgün et al., 2003] as social phenomena (See Chapter 4 for more details).

In principle, psychology tends to focus on the understanding of human decision making process from a micro-analysis perspective. As a result, a cognitive approach pursuing the investigation of human mind internals without consideration of external influences [Neisser, 1976, Anderson, 2005] has become popular. In contrast, sociology attempts to comprehend human behaviour in a way of macro-analysis which takes into account additional factors such as social context or environmental settings in conjuction with the individual mind [Anderson, 2005, Bandura, 2001, Akgün et al., 2003]. Hence, this attitude in sociology leads to the view that the cognitive reasoning process ought not be seen as sufficient to illustrate the mechanism of human choices, in particular under social settings. Instead, the socio-cognitive perspective has been given more attention for the better identification of underlying mechanism of human social behaviour [Granovetter, 1985, Stein, 1993].

In accordance with this perspective, this research centres on the provision of a computaional model facilitating agent reasoning behaviour with norms on the basis of a socio-cognitive perspective of human choice. To this end, we propose **DNA**^{3} based on the hybrid approach, which is a combination of the institutional model (also called normative frameworks) [Cliffe et al., 2009] as a means to achieve social cognition and the (BDI type) cognitive agent architecture as a means to achieve individual cognition. We start this work from the design of an integration middleware, BSF. Then we connect both BDI-type cognitive agents, ***N-Jason***, and institutions with virtual characters (SecondLife avatars in this case) into one system by BSF. It is followed by three experimental evaluations which design and simulate polite behaviours subject to norms between individuals and groups, with consideration of emergency situations. The details are as follows:

- **Coupling Cognitive Agents and Virtual Environments** We firstly do coupling BDI-type cognitive agents and virtual environments. This enables the control of virtual character behaviour by intelligent agents, specifically by goal-oriented agents in this thesis. Also, this promises the cognitive reasoning capability in

individual behaviour planning subject to the observation from virtual environments. We propose a middleware approach (via BSF) to integrate both softwares for this purpose.

- **Facilitating Automated Social Reasoning to Govern IVAs Behaviour with Norms** Given the above AI programming environment for IVAs, we integrate institutions being capable of social reasoning on top of the framework via BSF. This completes the **DNA**^{3} environment and enables the performance of social and individual reasoning simultaneously as soon as virtual characters broadcast percepts can be observed in the virtual environment. The final choice of plans to execute is carried out during the individual reasoning step in ***N-Jason***. When action plans are determined by the ***N-Jason*** agent, those plans are received and acted upon immediately by the virtual characters.

- **Norm Aware Decision Making in *N-Jason*** Although **DNA**^{3} allows the formulation and communication of norms with virtual characters so each IVA has explicit information in its mind, the actual deliberation inside agents still requires the internal mechanism to choose norms and agents' own goals. To this end, we apply the deadline and priority based scheduling algorithm [Alechina et al., 2012] adapted for ***Jason*** BDI agents in order to evaluate the importance and urgency of each intention that is the means-ends to achieve goals.

- **Evaluation** Given **DNA**^{3}, we aim to show how the appropriate behaviour of virtual characters can be exhibited by the combination of social and norm-ware cognitive reasoning. In here, three case studies on modelling politeness in virtual characters behaviour are presented as a means to verify the adequacy of **DNA**^{3} in the design and simulation of norm-awareness for virtual characters. We specifically would like to show the characteristics of **DNA**^{3} with respect to the flexibility in reasoning about norms subjec to changes of situaions and the rationality in norm-ware decision making in these scenarios.

1.5 Thesis Structure

This thesis is organised as follows:

Chapter 1 – Introduction This chapter introduces the main objective of the thesis and identifies the challenges we aim to address. Further, the contribution of the work is emphasised and an outline of the thesis as a whole is presented.

Chapter 2 – Related Works This chapter presents the background of the research we conducted to establish the computational model, ***N-Jason***, and reviews the literatures which illustrates several approaches to the use of norms in the virtual character behaviour.

Chapter 3 – A Middleware Approach to Connect Cognitive Agents to Virtual Environments This chapter presents an illustration of coupling BDI-type cognitive agents and virtual characters via the middleware approach, as a starting point of building norm-aware virtual characters. After the introduction to the integration middleware, BSF, we describe in depth an illustration of the integration through an example.

Chapter 4 – Governing Intelligent Virtual Agent Behaviour with Norms In this chapter, we present a computational model of the socio-cognitive theory on human choices, **DNA**^{3}, which is established by the integration of normative framework (or institutions), BDI-type cognitive agents and virtual characters. After a short introduction to socio-cognitive theory, the institutional model and its associated automated norm reasoning mechanism is presented. Afterwards, we provide an informal explanation of **DNA**^{3}which illustrates: (i) How reasoning about situationally appropriate norms is carried out and, (ii) How norms can be adopted in the agent's mind.

Chapter 5 – Norm Aware Decision Making in BDI Agents This chapter introduces the norm-aware cognitive decision making process in ***N-Jason*** (i.e. how norms can be associated with the agent decision making process) for reasoning plans to execute, particularly through the deliberation on norms and agents' individual goals. Firstly, we describe the extended features over the conventional BDI architecture and programming language in order to support normative concepts, then we propose norm-aware deliberation as well as run-time norm execution.

Chapter 6 – Case Studies This chapter presents the illustrative examples of applications of **DNA**^{3}. We particularly focus on the modelling of scenarios already achieved using both Decision Theoretic Models and Dynamic Planning models, in order better to contrast them with our approach. Firstly, we describe the polite behaviour between individual NPCs and PCs, then we extend this politeness model to inter-group behaviour. In addition, we simulate NPC behaviour when an emergency situation occurs. We in-

vestigate the performance with regards to the speed of taking actions and success rate of the tasks the NPCs have to achieve.

Chapter 7 – Conclusion This chapter wraps up the thesis, addressing the discussions including advantages and drawbacks of our approach, conclusion and future works in relation to those limitations.

1.6 List of Publications

The followings are papers all published by the author which are basis of this thesis. In each case the contribution of authors to the paper are stated in accordance with regulation 16.1 subsection 3.v of University of Bath regulations.

1. Run-Time Norm Compliance in BDI Agents - JeeHang Lee, Julian Padget, Brian Logan, Daniela Dybalova and Natasha Alechina. Published in the Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems (AAMAS '14). International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, pages 1581-1582.

2. ***N-Jason***: Run-Time Norm Compliance in AgentSpeak(L) - JeeHang Lee, Julian Padget, Brian Logan, Daniela Dybalova and Natasha Alechina. In Dalpiaz, F., Dix, J., and van Riemsdijk, M., editors, Engineering Multi-Agent Systems, volume 8758 of Lecture Notes in Computer Science, pages 367-387. Springer International Publishing. This is the full version of the AAMAS 2014 paper.

   Publications 1 and 2 introduce the norm-aware decision making through ***N-Jason***. Parts of these publications appear in Chapter 5.

3. A Semantic Approach to Situational Awareness for Intelligent Virtual Agents - Surangika Ranathuga, Stephen Cranefield, Julian Padget, JeeHang Lee. Submitted to the Journal of Autonomous Agents and Multi-Agent Systems.

   The main objective of publication 3 is the provision of situational awareness for virtual character behaviour by means of Complex Event Recognition (CER) on top of the combination of IVAs and normative frameworks. Prerequisites for this approach are the integration of multiple software components (e.g. cognitive agents, virtual characters, normative frameworks and event recognition systems)

on the one hand, and frequent exchanges of events with their semantics interpreted by CERs amongst them on the other hand. We contribute to this publication through the use of the Bath-Sensor-Framework as an integration middleware, which is also capable of the semantic representation of data to this end. Part of this publication appears in Chapter 3.

4. Towards Polite Virtual Agents using Social Reasoning Techniques - JeeHang Lee, Tingting Li and Julian Padget. Published in the International Journal of Computer Animation and Virtual Worlds. 24(3-4):335-343, 2013

   Publication 4 describes the polite behaviour of IVAs during individual navigation, which is expressed and realized using **DNA**^{3}. This publication forms the major part of Section 6.2 in Chapter 6.

5. Governing Intelligent Virtual Agent Behaviour with Norms - JeeHang Lee, Tingting Li, Marina De Vos and Julian Padget. Published in Proceedings of the 2013 international conference on Autonomous agents and multi-agent systems (AAMAS '13). International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, pages 1205-1206.

6. Using Social Institution to Guide Virtual Agent Beahviour - JeeHang Lee, Tingting Li, Marina De Vos and Julian Padget. Presented at the 2^{nd} International Workshop on Cognitive Agents for Virtual Environments (CAVE2013) at AAMAS2013 - Saint Paul, MN, USA, 2013. This is the full version of AAMAS 2013 paper.

   Publication 5 and its full version, item 6 introduce **DNA**^{3}, a computational model of a socio-cognitive theory of human choices, comprising the integration of ***N-Jason*** agents, virtual characters and normative frameworks. Parts of these publications appear in Chapter 4.

7. Decoupling Cognitive Agents and Virtual Environments - JeeHang Lee, Vicent Baines and Julian Padget. Published in Cognitive Agents and Virtual Environments, Lecture Notes in Computer Science Volume 7764, pages 17-36, 2013. Presented in CAVE 2012 in Valencia, Spain

   Publication 7 introduces an integration middleware, Bath-Sensor-Framework (BSF), and then presents a middleware approach to connect ***N-Jason*** agents and virtual characters. Part of this publication appears in Chapter 3.

Chapter 2

Related Works

In this chapter, we discuss literature describing state–of–the–art in the domain of intelligent virtual characters that consider social norms as part of their behaviour. As mentioned in Chapter 1, the decision making is usually achieved by three types of agent models – Reflex, Goal-based and Utility-based agents [Russell and Norvig, 2002]. Here, we review research implementing norm-aware behaviour in virtual characters through these types of agents.

2.1 Introduction

Advances in psychology theories contributing to understanding the mechanisms of how human brain works, lead to the advent of various computational models in the domain of AI. These computational models in turn have encouraged AI researchers to pay much attention to agent behaviour in response to sensing the environment in which it is situated as a means to replicate/simulate human intelligence and behaviour.

Usually, intelligent agents (e.g. reactive, decision theoretic or cognitive) [Group, b, Fikes et al., 1972, Brooks, 1991, Brooks, 2001, Bryson, 2003, Brom and Bryson, 2006, Rao and Georgeff, 1995, Mascardi et al., 2005] analyse this information, make decisions, and do planning for the next behaviour to execute. All of these approaches effectively take the position that the entirety of the perceivable knowledge, possibly even its (partial) history and the whole of the decision-making process are internal to the agent, in order to make the system self-contained and perhaps also better to reflect the notion of an independent intelligent entity.

Awareness of social norms can be added to any agent model, although it is an open issue whether norms should be an external source of knowledge that delivers obligations (also called norms in some literature) to agents [Alechina et al., 2012, Meneguzzi and Luck, 2009], or be internalized in some way and then whether they are absorbed into the agent reasoning process [Andrighetto et al., 2010] or kept separate so they can be 'switched off' when no longer applicable [Criado et al., 2010]. These issues aside, it is clear that the addition of social norms into intelligent agents is viewed as one way in which to enhance agent reasoning and response capabilities and in particular to enhance response in social settings. As we discussed earlier in Chapter 1.1 with observational studies [Yee et al., 2007, Samperi et al., 2012, Friedman et al., 2007], almost all social behaviours occurring in virtual worlds by human users are governed by the same social norms as those in the real human society. This implies that if we wish the behaviour of IVAs to resemble that of humans more closely, then it is desirable somehow to convey those real world human social norms to the agents.

What we firstly describe is a single agent perspective that social norms are internalised in agents' mental states (Section 2.2) on the one hand. Norm-awareness in virtual characters behaviour achieved by reflex agents (Section 2.2.1) and utility based agents (Section 2.2.2) is presented in here. Afterwards, we introduce the Multi-Agent Systems (MAS) approach that social norms are external knowledge and delivered to the agent's mind to guide virtual character behaviour (Section 2.3) on the other hand. The summary is presented at the end with an evaluation of pros and cons of each approach compared to our proposal, **DNA**^{3}, a socio-cognitive computational model.

2.2 Norms in a Single Agent

2.2.1 Reactive Approach

The underlying principle of reflex agents in decision making is action selection in a reactive manner. Only the percepts that agents currently observe affect the process of action selection. When agents perceive events from an external environments, reflex agents immediately search for a corresponding plan dedicated to the symbolic representation of those external events, and execute a single selected plan reactively in consequence [Russell and Norvig, 2002]. The most famous reactive architecture is

sub-sumption [Brooks, 2001]. In this architecture, agent behaviour is represented as reactive plans in the form of an *'if-then'* structure. Those reactive plans are hierarchically organized by relative priority between the plans. Finite State Machines (FSMs) are also popular in the design of reactive plans, where the effect of current conditions brings about a new state. Beyond the architectural approach, this type of agent can be developed as simple conditional branches in ordinary high-level programming languages. In accordance with such principle, the social norms that virtual agents implicitly possess are in the form of reactive plans.

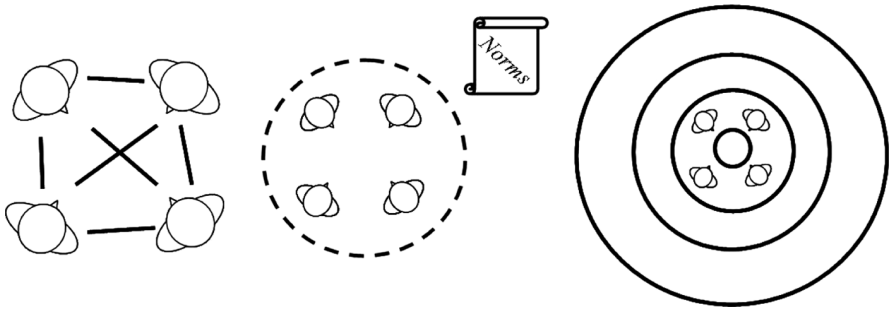

Figure 2-1: *Conversations and Human Territories. (c) Pedica et al., 2010. [Pedica and Högni Vilhjélmsson, 2010]*

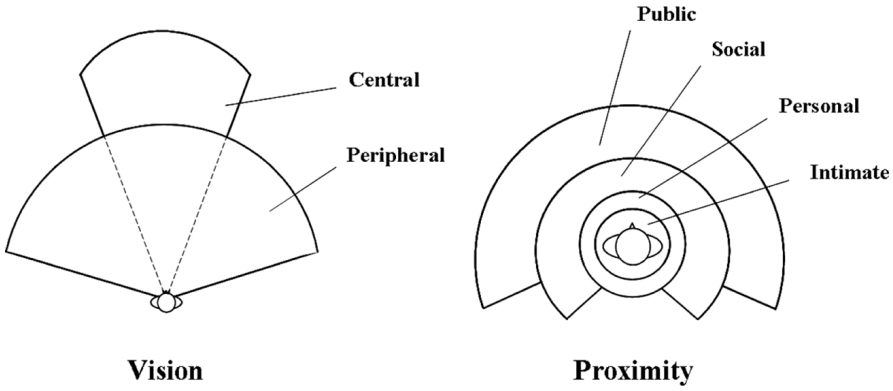

Figure 2-2: *The paradigm of territorial organisation. (c) Pedica et al., 2010. [Pedica and Högni Vilhjélmsson, 2010]*

Pedica *et al.* [Pedica and Högni Vilhjélmsson, 2010] take this approach; social norms are represented as reactive plans in reflex agents, in order to generate socially believable behaviour of virtual characters. They in particular concentrate on the exhibition of avatar behaviour with regards to dynamics in social interactions. Inspired by theories on human territoriality [Scheflen and Ashcraft, 1976, Kendon, 1990], this

research models and simulates the territorial dynamics of social interactions (e.g. conversational situations, see Figure 2-1) by means of a set of social norms. These norms indicate reactive motions defined in the territorial organisation (see Figure 2-2) subject to proximity and social intimacy between virtual characters. In other words, social norms address the territorial rights and responsibilities of participants (who are currently engaged in a specific group) and others (such as new-comers or outsiders from the group who are willing to join). Depending upon a personal or organisational relationship, avatars can take part in the interactions within a spatial boundry the group allows, or just exhibit conventional social behaviour apart from the group.

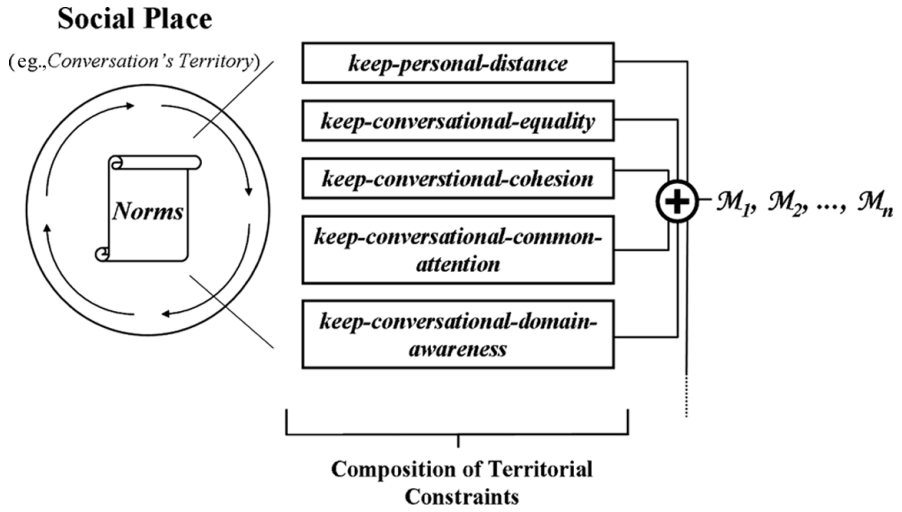

Figure 2-3: *Models of social norms. (c) Pedica et al., 2010. [Pedica and Högni Vilhjélmsson, 2010]*

The models of social norms by which social awareness of virtual characters is achieved is shown in Figure 2-3. In this system, avatars behaviour is realised by the activated set of reactive plans which actually embody a norm determined by a social situation (i.e. social place) in which avatars are situated. This activation of reactive plans gives rise to the group dynamics a group of people are now expecting.

All of this process is determined by a Reaction Generation Framework described in Figure 2-4. In essence, this framework pursues *"reactive response"* rather than a deliberative reasoning process, such as that which cognitive agents usually perform for the purpose of a quick response to dynamic changes in virtual environments. Thus, the internal process is intuitive but simple: when an agent perceives a low-level information, this is analysed by a set of reactive plans. Afterwards, the outcome of analysis triggers

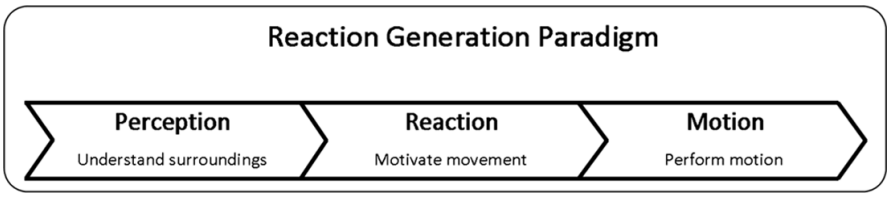

Figure 2-4: *Reaction Generation. (c) Pedica et al., 2010. [Pedica and Högni Vilhjélmsson, 2010]*

an immediate motion (i.e. low level implementation such as to produce an animation in response to low-level perceptions).

In conclusion, the use of reflex agents which enables social norms through reactive plan, promises: (i) a good performance with regards to response time and (ii) a simplicity in design of social norms and its associated behaviour as Pedica *et al.* emphasise. Despite these advantages, there is a downside of this approach. First, it seems hard to say that social norms described in the form of reactive plans are formal representations of norms although the designer of norm-aware reflex agents claims that it makes sense ontologically. Those social norms are likely to be a form of ordinary reactive plans at a syntactic level rather than a formal model containing the essential properties of norms such as normative position, deadline, priority or sanctions. Thus it is hard to show more delicate model of norm compliance since those properties can not be considered. In addition, agents cannot violate those social norms, thus norm-autonomy is bounded since compliance with all social norms is already preordained inside an agent's mental states. Lastly, agents may not be able to show social norm aware behaviour when the environment or a situation (in terms of place and time) is changed. Since those norms are pre-built knowledge and agents are not able to reason about norms subject to changes in situations, this type of agent cannot always show a proper level of social believability as Pedica *et al.* aim to.

2.2.2 Utility Based Approach

In principle, a central concept of utility based agents is to pursue the highest satisfaction or optimality in decision making. Once an utility based agent adopts a goal, its reasoning process tries to find a specific combination of actions to achieve the goal, which ensures the maximisation of utility. Each action (or step) has a real number which is mapped to a degree of satisfaction or optimality (or sometimes called weight), when

an agent takes such action on the way to achieving a goal. The representative example is a decision theoretic agent framework which combinies probability and utility theory (e.g. probability distribution of candidate actions agents execute to achieve a goal in environments). The agent updates its current state based on beliefs (updated by perceptions) and a way of choosing an action with highest expected value under the current context [Russell and Norvig, 2002].

Thespian [Si et al., 2006] is a decision theoretic framework that facilitates modelling of social norms and its associated behaviour for virtual characters controlled by utility-based agents. In this work, Si *et al.* aim to develop a human-like virtual character which is capable of holding a conversation with human characters as social norms governing conversational situations in the real world. On top of Partially Observable Markov Decision Processes (POMDP) [Smallwood and Sondik, 1973] enabling an alternative to the explicit representation of norms, the Thespian system also allows virtual agents not only to have explicit goals to comply with social norms and personal goals, but also to reason about those normative and personal goals to avoid conflicts between them. The reasoning about conflicts is in turn driven by evaluating the effect of achieving or dropping personal goals in response to complying or violating social norms.

In essence, a Thespian agent has: (i) goals which trigger the correct behaviour in a social context, (ii) state features[1] that the agent pursues to maximise or minimise (e.g. the status of conversation, affinity between characters and obligations each character possesses) and (iii) dynamic functions which update state features by defining how actions/beliefs can affect agent's states. Virtual characters mainly seek to achieve goals with intention of maximising all those state features using dynamic functions. The final decision is determined by measuring the weight of each goal in the virtual character.

Table 2.1 shows a model of conversational norms used in this system. In addition, the algorithm of dynamic function in relation to Table 2.1 is illustrated in Algorithm 1 as an example. This algorithm shows the process that enforces those adjacency pairs in a virtual character's behaviour by triggering a goal of each agent with a request to maximise the state feature, *complete_adjacency_pair_norm*. Depending upon the state feature in the agent's mind, the best choice can be either violation of or compliance with the norms

[1]State is a set of state features which are agent properties such as name, age and affinity between two characters.

Speaker 1	**Speaker 2**	**Obligation**
Greet	Greet back	Greet back to *speaker 1*
Bye	Bye	Say "Bye" to *speaker 1*
Thanks	You are welcome	Say "You are welcome" to *speaker 1*
Offer *X*	Accept/Reject *X*	Either accept or reject *X* to *speaker 1*
Request *X*	Accept/Reject *X*	Either accept or reject *X* to *speaker 1*
Enquiry about *X*	Inform about *X*	Inform to *speaker 1* about *X*
Inform information	Acknowledgement	Acknowledgement to *speaker 1*

Table 2.1: *Adjacency Pairs and Corresponding Obligations. (c) Si et al., 2006. [Si et al., 2006]*

Algorithm 1 Dynamics for complete_adjacency_pair_norm. (c) Si *et al.*, 2006. [Si et al., 2006]

```
1: if self == dialogueact.speaker then
2:     if dialogueact intends to satisfy an obligation then
3:         if the agent has such obligation then
4:             return original_value + 0.1
5:         else
6:             return original_value - 0.5
7:         end if
8:     end if
9: end if
10: return original_value
```

The use of utility-based agents could be a good choice to engineeer norm-aware virtual characters with regards to the norm autonomy of virtual characters. This type of agent is able to provide an opportunity to violate norms subject to the utility the violation brings about, while reflex agents must comply with norms as reactive plans enforce them without further consideration. By promising a means to evalute the importance (or weight) of each behaviour triggered either social norms or individual goals, a rational deliberation on norms and goals can be determined in utility-based agents.

However, the use of utility-based agents shows a couple of shortcomings. On the one hand, reasoning about norms depending on dynamic changes in a current situation is not possible in this setting. As described earlier, social norms are already pre-defined in an agent mind with a relative imporance (represented in a real number). The observation of changes in the environment by an agent can only trigger situation-specific norms, in turn leading to the calculation of an utility in order to carry out decision mak-

ing for the next behaviour. During this process, the primitive role of observations is not to bring about norms subject to a new situation the agent encounter, but to change the agent mental states so as to adopt associated social norms specified by preordained rules.

On the other hand, decision theoretic frameworks (e.g. probability or utility distributions of norms, actions and behaviour) show a low level of robustness with regards to environmental settings. In other words, those distributions and reasoning dynamics are closed dedicated to the specific environment and thus utility-based agents cannot exhibit correct behaviour when the agents are situated in a totally different environments, where desirable behaviour may differ from the place where the distributions are designed.

2.3 Norms as an External Knowledge in MAS

What we introduce in this section is an approach to using Normative Multi-Agent Systems (NorMAS) with an external source of situationally adequate norms in order to guide virtual character behaviour. We firstly give an overview of NorMAS which is a combination of normative frameworks and MAS. This includes some background on norms, normative systems in MAS, and a brief description of governance mechanisms for individual agent behaviour in NorMAS. Afterwards, we present an in-depth survey of norm-aware behaviour of virtual characters residing in NorMAS. For this, we show some examples following (i) regimenation approach that normative frameworks directly governs virtual characters behaviour and (ii) regulative approach that normative frameworks privides a set of norms as a guidance, and the final decision is determined by the agents.

2.3.1 Normative Multi-Agent Systems

Multi-Agent Systems (MAS) are a societal structure where autonomous agents are residing and taking actions and interactions. While the aim of individual agents is to achieve a task by both reactive and proactive behaviour in an agent society, the ultimate objective of MAS is to coordinate and regulate individual agents so as to achieve desirable goals or states in the agent society incorporated with social interactions between

agents. This nature of MAS reminds us the similarity to the human society which is defined as "*Society merely is the name for a number of individuals, connected by interaction*" [Simmel and Wolff, 1950]. As mentioned in Chapter 1, norms lead people to have enhanced social awareness, which in turn play a crucial role in human choices during human interactions. Likewise, it is feasible that computationally well formalised norms for MAS can also shape and regulate individual agents' behaviour, specifically with respect to socially acceptable actions and interactions in an agent society.

As a result, MAS has given much attention to the incorporation of norms in agent behaviour. This trend in consequence results in a facilitation of Normative Multi-Agent Systems (NorMAS). In essence, the central concept of NorMAS is to organise the normative behaviour of individual agents in conjunction with an external source of norms and/or organisational rules in MAS. Typically, NorMAS is composed of (i) a normative framework and (ii) a MAS.

The normative framework is the main entity for the governance of individual agents' behaviour in MAS. Their central role is to specify and reason about norms in response to both external events both that agents bring about and changes of state in the environment. Once situationally correct/required norms are determined by this framework and in turn broadcast to MAS, then individual agents adopt those norms and start the practical reasoning process in response to determine whether to comply or not with those norms. In some cases, those norms enforce or regiment individual agents' behaviour via monitoring and sanctioning mechanisms for the social purpose. Given these characteristics, NorMAS provides a good means for coordination and cooperation, group decision making, regulation of agent societies and so forth in the domain of MAS researches [Boella et al., 2006].

Norms in NorMAS

In NorMAS, the primary entity for regulating an agent society is normative frameworks (e.g. institutions, organisations). The normative framework is a set of rules which indicate situation specific norms and organisational rules. Actually, computationally formalised norms in these systems are classified as *Constitutive* and *Regulative* norms [Therborn, 2002].

Constitutive norms, on the one hand, are the categorical property of normative systems, which stand for the definition of functions of the normative system and the mem-

bership of an agent participating in that normative system [Therborn, 2002]. The constitutive norms regulate not only the creation of institutional facts but also revision of the normative system itself. Based upon this concept, the consitutive norms act as a belief prescribing a category of the normative system as well as specifying behaviour of the normative system [Boella and van der Torre, 2004]. On the other hand, regulative norms typically address an action or event the system requires the participants to carry out in response to the current state of the system. In other words, the regulative norms directly indicate a normative position of a particular action/event such as obligations, prohibitions and permissions which should be achieved by agents that the normative system governs. Thus, the regulative norms are able to act as desirable goals (or states) the normative system is currently pursuing [Boella and van der Torre, 2004].

Depending on the formalism of the normative system, the regulative norms can be categorised as *state-based norms* or *event-based norms*. State-based norms usually express higher level norms that impose desirable or required states on the system (or an environment), often as a logical combination of institutional facts, which should be brought about by the actions of agents (e.g. agent i is obliged to bring about a state ϕ) [Dignum, 2004]. In contrast, event-based norms generally represent relatively lower level activities addressing possibly executable events (or actions) at the individual agent level (e.g. agent i is obliged to perform an action α) [De Vos et al., 2013].

The Life Cycle of Norms in NorMAS

In NorMAS, norms are generally created by a set of normative multi-agents in conjuction with normative frameworks. These newly created (potential) norms will become (actual) norms not only in agents' mind but also in MAS, through a chain of processes known as '*norm emergence*'. Transmission, enforcement and internalisation of norms are major sub-processes of norm emergence [Hollander and Wu, 2011].

Once new (potential) norms are created by reasoning about norms (which usually takes place in normative agents and/or normative framework), in response to events they observe from the environment, the (potential) norms can spread from agents to agents, or from normative frameworks to agents, by various transmission mechanisms such as broadcasting, publish/subscribe or other protocols. If some agents are exposed to those transmitted (potential) norms, the enforcement process ensures the acquisition of those norms in the agents' mind. Then, the agents begin to shift its preference from

the original set of (actual) norms to newly incoming (potential) norms depending upon personality, desire or contextual knowledge of the agents. At some point a sufficiently large number of agents change their preference to the newly created (potential) norms, these new (potential) norms are internalised, thus becoming (actual) norms, which are able to trigger normative goals or desirable normative states to achieve [Hollander and Wu, 2011].

In the end, the context of the environment in which multi-agents are situated has changed, thus it would be undesirable to comply with the specific (actual) norms in agents' mind. In other words, a particular set of (actual) norms can be no longer valid under the changes of circumstances. Then, these (actual) norms become candidates which should be forgotten, and new (potential) norms may be created via the continuation of the new cycle of normative emgergence process [Hollander and Wu, 2011].

NorMAS and Agent Behaviour

As we have seen in the earlier section (Section 2.3.1), NorMAS is a combination of normative framework and MAS. As indicated in the norm emergence process, the normative framework (e.g. institution or organisation) is in charge of governing the behaviour of individual agents by means of social norms. These norms are specified in the frameworks, and in turn determined by taking into account external events subject to the context of individual agents (e.g. situations or roles). This results in a detachment of a new set of (social) norms which influence individual agents' behaviour. Normally, the new set of norms is delivered to an agent's mind via communication (e.g. by broadcasting or a specific protocols). Once agents adopt those norms in their mind, agents start a practical reasoning process about whether to comply with those norms or not depending on the mechanism that determines agent behaviour in response to a norm.

The implementation of this governace has broadly taken one of two approaches: regimentation or regulation. The former requires total compliance of the agent with the norms – so they are no longer norm autonomous and violation in turn is not possible – whereas the latter provides compliance information to the agent, but the decision whether to comply or not remains with the agent – norm autonomy might be retained, but violation is possible.

The regimentation approach is developed in many works [Boman, 1999, Verhagen and Boman, 1999, Esteva, 2003, Bogdanovych et al., 2008b]. The underlying mecha-

nism of the regimentation is the direct control of agent behaviour by means of norms detached by normative frameworks. In this approach, the choice of required/desirable behaviour in the current situation is done by normative framework. In accordance with this, individual agents just simply follow the specified normative behaviour without further deliberation on norms and individual goals when norms are detached. Thus, the main role of agents is effectively limited to perceiving external events and realising actions in the environment.

Electronic Institution (EI) is a representative example of an institution based upon regimentation, in which heterogeneous agents including humans and software entities can participate by playing different roles and can interact via speech acts [Esteva, 2003]. The EI defines a set of constraints, such as what participants are permitted or forbidden to do depending upon roles the participants have already taken. Each role identifies activities that agents should do in the current situation (which is referred to as a *Scene* in the EI). Reasoning about norms is carried out by given transition rules in the *Scene*, which identify consequence between contextual knowledge (e.g. external events agents observations) and norms (e.g. activities in roles). As soon as norms are detached, agents just simply obey these norms without further deliberation on norms and agent's individual goals. All that is allowed to agents is a total compliance of norms from EI thus no violations occurs under this context.

In contrast, the regulation (or governance) approach, explored in [Dignum, 1999, Boella et al., 1999, Hogg and Jennings, 2000, Falcone and Castelfranchi, 2001, Broersen et al., 2002, Cliffe et al., 2007], allows agents to have autonomy with respect to norms. In other words, norms can be both accepted and violated by agents depending upon their own situation. This approach usually takes a combination of normative frameworks and goal-oriented agents (e.g. cognitive agents) to produce a norm compliant behaviour in agents. When agents perceive external events or changes of state in environments, the observations trigger reasoning about norms in the normative framework. As soon as this finishes, the agent adopts (a set of) norms into its mind and performs individual reasoning. Depending upon the preordained rules in the agent programs of a goal-oriented agents, norms can either be a goal itself that the agent wants to pursue or be a part of a mental state which can trigger a goal or plan to achieve a state/action in the current situation.

The goal-generative approach [Dignum, 1999, Falcone and Castelfranchi, 2001,

Broersen et al., 2002] is a good example. In this approach, norms defined explicitly are coped with as one of the influential elements in the reasoning process of cognitive agents. Norms are not hard-wired into agents but can be acquired by agents through interactions with other agents which have different norms [Savarimuthu et al., 2008]. These norms are likely to be used in the process of decision making, either for the plan selection by preference ordering of norms [Dignum, 1999, Falcone and Castelfranchi, 2001], or for the goal generation with belief, intention, obligation and desire [Broersen et al., 2002]. This is very important aspect for norm-aware intelligent agents, because not only can agents choose the fact whether norms might be violated or not, but also norms might not always be necessary for agent decision making. This means norms can be either totally neglected, partly considered or fully affective for the final decision, depending upon the situational information of the agent.

2.3.2 Virtual Characters Behaviour in NorMAS

As mentioned earlier, taking norms into account in determining agent behaviour can be categorised as either regimentation and regulation. The domain of virtual characters also take these disciplines in the design and simulation of norm-aware behaviour. We firstly introduce a regimentation approach in this domain, which is usually implemented by the combination of normative framework and virtual character with no reasoning capability. Afterwards, a regulation approach is presented, which is composed of normative framework, virtual characters and reasoning agents (usually goal-oriented agents).

Regimentation of Virtual Characters Behaviour

Virtual Institution (VI) [Bogdanovych, 2007] is pioneering research on facilitating NorMAS in Virtual Worlds (VWs) on the basis of regimentation approach. The VI aims to construct 3D EI (which is a combination of EI and VWs) to be able to control participants' actions and interactions based upon norms and associated behaviour. This approach aims to embed EIs into Virtual Worlds (VWs), in particular in the physical 3D spaces such as building, rooms or halls where interactions may be able to take place between Player Characters (PCs) and Non-Player Characters (NPCs). During the interactions in a specific 3D space, EIs embedded in that place recommend and control

characters behaviour in a same way as that described in the earlier section (Section 2.3.1).

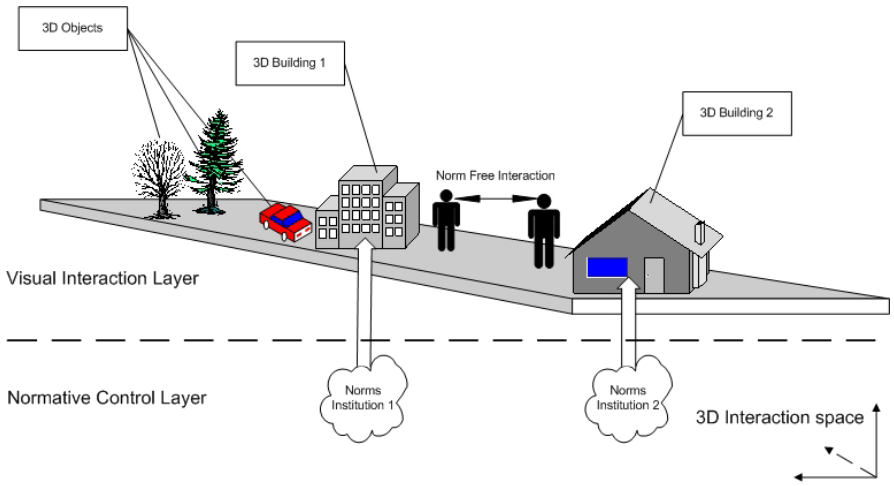

Figure 2-5: *Basic Concept of Virtual Institution. (c) Bogdanovych, 2007. [Bogdanovych, 2007]*

Figure 2-5 shows the basic conception of VI. The EIs are represented as a 3D building and its internal spaces (e.g. rooms, halls) in the VW. In other words, each institution is mapped to each 3-dimensionally visualised physical space. This pair of EI and 3D virtual space is called a 3D EI. As we see the Figure 2-5, they divide the 3D virtual world into a set of 3D physical places such as 3D object, rooms and buildings in Visual Interaction Layer (VIL). The individual physical place is regarded as an 'EI' described in the earlier section (Section 2.3.1). Inside a 3D EI (e.g. building), PCs and NPCs can take roles and carry out activities allowed by the EI. Of course, EI specifications and its reasoning about normative actions and interactions are managed in Normative Control Layer (NCL).

From the engineering perspective, the system is deployed by the integration of three-layers described in Figure 2-6. The *Visualization Layer* is used for the graphical visualisation of 3D VWs. Not only actions and interactions of avatars but also graphical representation of 3D environments can be displayed by this layer. In addition, a detection of activities which is essential to normative reasoning is also taking place in this area. The *Communication Layer* is in charge of communication as well as integration between VWs and EIs. When virtual character activity and its influence on VWs are detected, this information is transferred by this layer. In the opposite direction, when changes of institutional states (e.g. norms) result from those information observed in

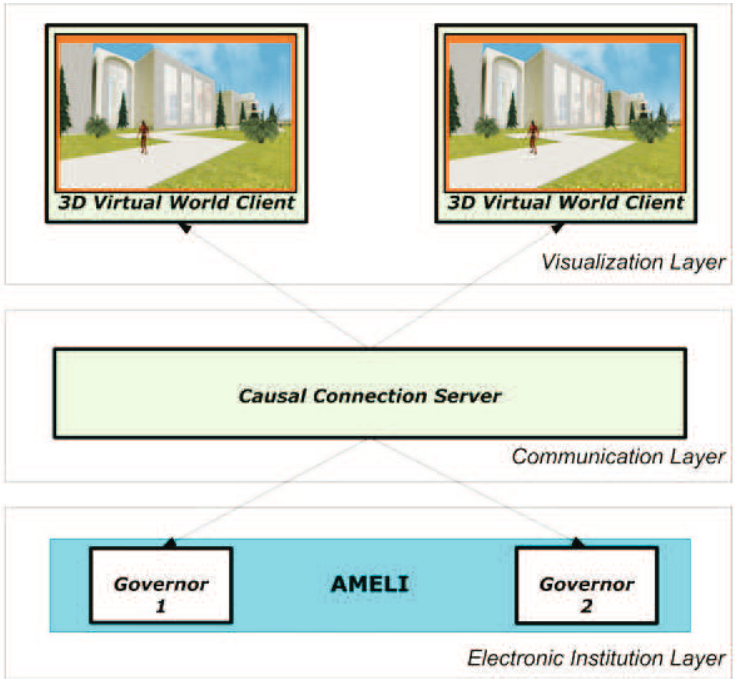

Figure 2-6: *Runtime Model. (c) Bogdanovych, 2007. [Bogdanovych, 2007]*

Visualization Layer, these changes of states are also transferred via the causal connections server in *Communication Layer*. Reasoning about and inspection of normative behaviour in response to environmental changes that the virtual characters bring about is carried out by *Electronic Institution Layer*, which is established by AMELI [Arcos et al., 2005]. This layer holds the institutional states and uses these states in conjunction with the institutional specification to make sure that NPCs cannot violate norms.

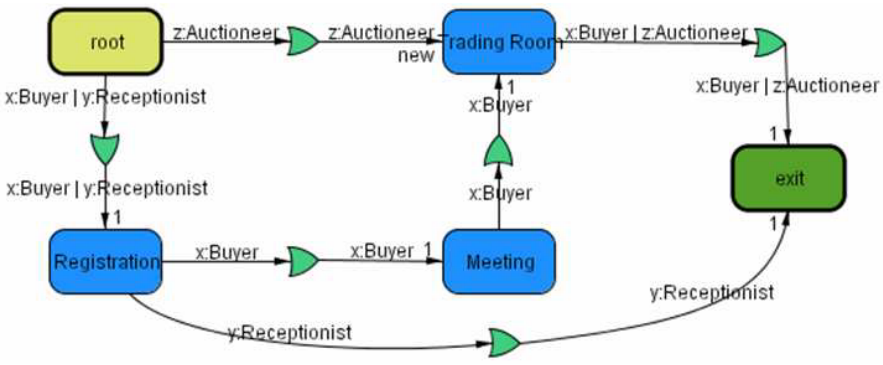

Figure 2-7: *Performative Structure. (c) Bogdanovych, 2008. [Bogdanovych et al., 2008a]*

The choice is actually determined through the performative structure in the *Elec-*

tronic Institution Layer. According to Figure 2-7, the performative structure consists of scenes where interactions between PCs and NPCs are articulated. Each scene defines a dialogical protocol in which virtual characters may be involved. With the definition of relationships in the performative structure, virtual characters are able to move from one scene to other scene depending upon both their role and observed external events. When NPCs are situated in a specific scene (e.g. rooms), this role and associated events in the full set of observations are engaged for the detachment of norms. As we discussed earlier, normative rules in the scene specify the consequence between characters contextual information and required norms. Thus, once NPCs perceive external events, the combination of these observations and its role trigger the reasoning about norms, and then the consequences of actions open to NPCs is determined by the normative rules.

As soon as NPCs adopt the consequence of actions, these actions would be a commitments that participating NPCs acquire and have to fulfill later. With regards to reasoning about NPCs behaviour, the deliberation is not taken place at an agent level in principle. As we can see, the determination of the behaviour is completely done by the EI in the 3D interaction space. Virtual characters are just requested to comply with norms through carrying out a course of actions either to fulfill norms or to achieve a normative state. In this sense, virtual characters behaviour is completely regimented by the request of EI. Without doubt, it is very hard to violate norms, therefore the norm autonomy of virtual characters is circumscribed.

This concept of VI has been popular as a engineering methodology to model and implement NPCs behaviour [Bogdanovych et al., 2008a]. In particular, this methodology is in the pursuit of interaction design between NPCs. The goal of this methodology is the provision of more *"believable"* actions/interactions of NPCs as PCs perform the same activities in VWs. Bogdanovych *et al.* take advantage of this methodology to produce e-Learning purposed serious games such as Virtual Heritage Application. In *The City of Uruk* project [Bogdanovych et al., 2009, Bogdanovych et al., 2012], much attention has been given to the design and simulation of the accurate illustration of everyday activities of ancient inhabitants (e.g. eating, sleeping, working or communications) with one another, whereas ordinary cultural heritage researches tend to focus on the realistic reconstruction of environments (e.g. buildings or artifacts). Not only are the daily activities of ancient characters displayed by roles defined in the EI, but also interactions between characters are automatically presented by reasoning about norms in accordance with the performative structure.

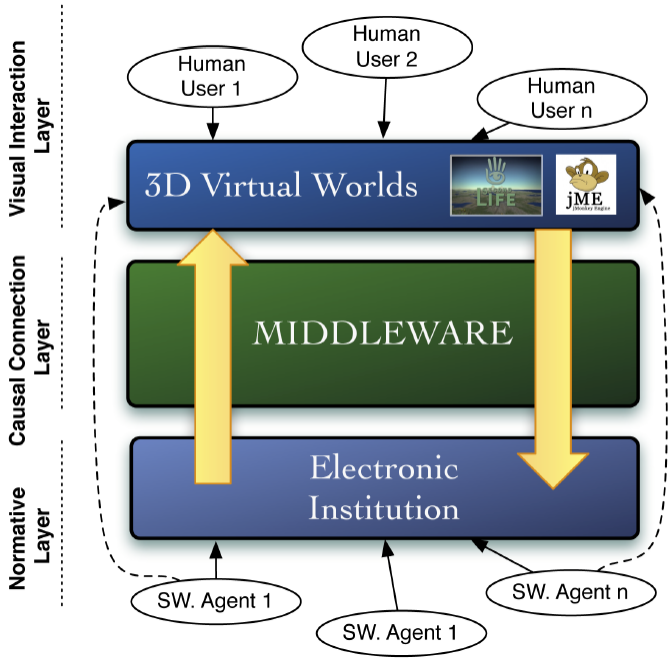

Figure 2-8: *VIXEE Architecture in v-mWater, (c) Trescak, 2011. [Trescak et al., 2011]*

Similar to this line of research, Almajano *et al.* [Almajano et al., 2013] propose an e-Government application which enables govenrmental administrative services for real humans by utilising the concept of VI. The representative example is *v-mWater*, which is a (virtual) governmental auction market for water services where the interactions between human users and autonomous virtual characters take place live. It is well known that auctions require specific interaction protocols to make an agreement for a certain transaction between participants. To this end, the VI methodology is employed to design formal specification of interactions which are essential for all procedures during the auction market. Interaction specifications are used in the normative behaviour reasoning for both NPCs and human users whereas Bogdanovych's e-Learning applications [Bogdanovych et al., 2009, Bogdanovych et al., 2012] are designed only for NPCs. In *v-mWater*, the outcome of determination of normative behaviour is detached to manipulated NPCs behaviour directly for NPCs. In contrast, this outcome is a guidance that clarifies required interaction procedures human users should take at the current situation for human users. Thus, human users are able to participate in the auctions and other services by looking at and complying with the guidance recommended by VI with higher adequacy in those interactions, even though the human users may have less (or no) idea about how to act in that place. This extended feature is facilitated by the

VIXEE architecture [Trescak et al., 2011] (Figure 2-8) which supports the integration of multiple virtual environments on top of VI.

The VI methodology is also employed to establish imitation learning environments for autonomous virtual characteres thanks to the nature of performative structure representing standard behaviour in social interactions [Bogdanovych et al., 2008c, Bogdanovych et al., 2008b]. In this case, the specifications in VI describing roles and its associated normative behaviour (e.g. scenes, rules and performative structures) can be regarded as a formal model of actions/interactions of virtual character behaviour. This formal model is mainly used as a means to interpret other characters actions. Let consider a trainee, who is the primary subject for learning by imitation, and training characters controlled either by human player or by agents capable of reasoning. Once the trainee observes a series of external events a trainer character brings about during interactions, the trainee starts a simulation of reasoning about normative behaviour with the observation (i.e. a series of external events) using the formal specification of VI. If this simulation is successful to bring about a set of norms, this set of norms in response to the observation (e.g. a sequence of external events used in the simulation) is taken by the trainee as a standard interaction pattern.

All in all, the VI and its applications, despite regimentation characterstics, is a positive demonstration of the benefits of integrating normative frameworks with VWs, showing how norms can affect participants' behaviour. However, there are some drawbacks to their approach. The most significant, in our view, is that agents do not have the possibility to reason about norm compliance: they are required to follow pre-defined behaviours without consideration of the current situation. Furthermore, these behaviours are tied to specific roles, so that an agent must choose to take on a role that was defined when the EI was specified. An agent can reason about its situation, but actions are restricted to those defined as norm-compliant at the design-time of the institution. Thus, the agents are subservient to the goals of the institutions and should only choose to surrender their autonomy for the period of institutional interaction, if the outcome aligns with their own goals. Furthermore, there appears to be no scope for adaptation to account for a changing environment.

Regulation of Virtual Characters Behaviour

One of the most important characteristics of intelligent agents is autonomy: agents should be able to take an initiative whether to pursue its own goals or not. Actually, this may imply that the autonomy is a capability of decision making on whether or not to achieve the goals triggered by either internal motivations or external influences. In other words, both goal autonomy (which is autonomy with respect to individual goals) and social autonomy (which is autonomy with respect to other actors) are essential to becoming an agent [Castelfranchi, 1995].

What the regulation approach ultimately seeks for virtual character behaviour with norms is to ensure this autonomy for the deliberation over both goals and norms as discussed earlier (Section 2.3.1). In other words, the regulation approach intends intelligent agents to be able to have more opportunities to make a decision about whether to comply with norms or not. For this reason, the regulation approach prefers the use of a reasoning agent (e.g. goal-oriented agents) to control its dedicated NPC, in conjunction with normative frameworks, as opposed to the direct control of NPCs behaviour by way of regimentation.

In general, the reasoning agent is coupled to a specific NPC. This pair of reasoning agent and NPC is likely to be seen as a set of *mind* and *body* of the virtual character: the mind thinks about what to do next based upon the current context, and the body not only observes external worlds but also performs actual movements in relation to the mind. With this paradigm, the reasoning agent becomes an essential means to determining plans to execute in response to percepts the NPC observes. These plans are executed by actuators the NPC realises. In NorMAS settings, the choice between goals and norms also takes place in the reasoning agent in the same manner. When a new set of norms is detached in response to NPC percepts, these new norms are likely to be an explicit goal itself with which an agent should comply or a certain condition (or state) to trigger an individual goal that an agent would like to achieve. As a result, the reasoning agent tries to make a decision between norm-generated goals and individual goals that ordinary percepts generate. The final decision might be made by several properties, or combinations of them, such as personality [Broersen et al., 2002], preferences [Alechina et al., 2012, Padgham and Singh, 2013] or emotion [Ferreira et al., 2012].

Ranathunga [Ranathunga, 2013] demonstrates virtual football players inspired by

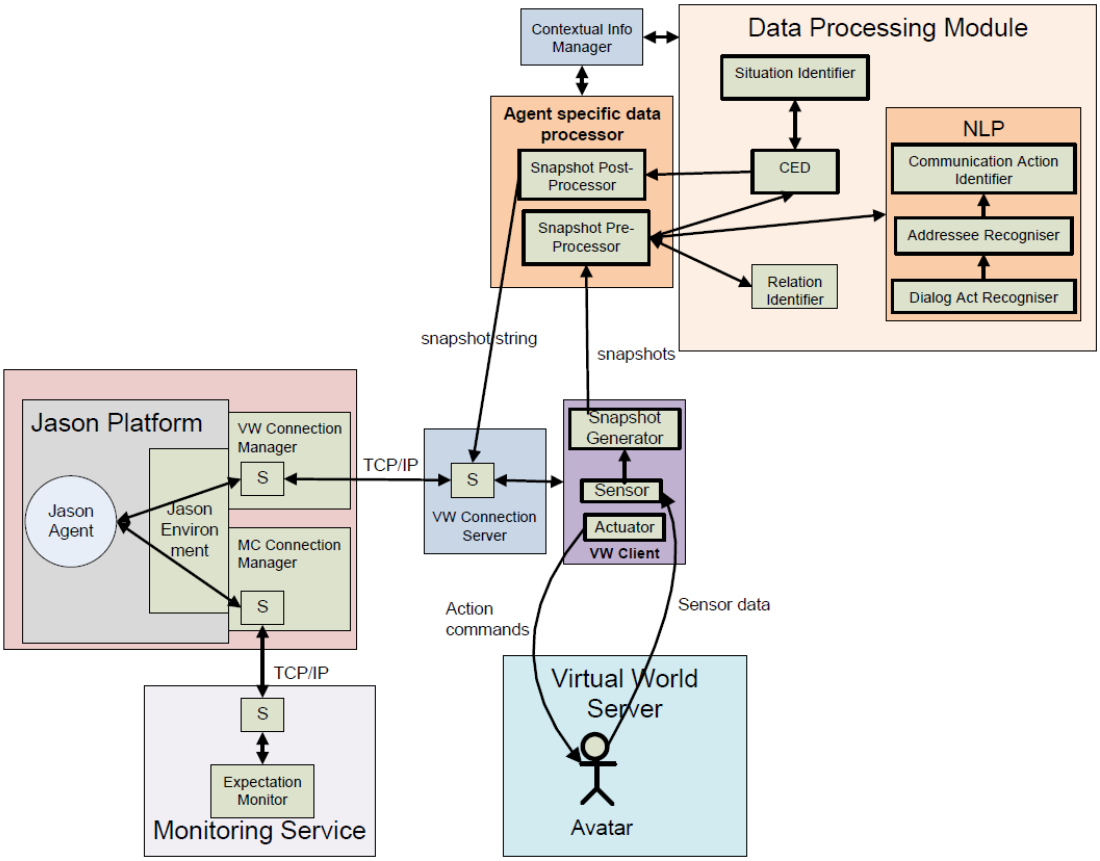

Figure 2-9: *Framework for Improving Physical and Social Awareness of an IVA. (c) Surangika, 2013. [Ranathunga, 2013]*

the regulation approach. In this work, a framework for improving physical and social awareness of IVAs is proposed in order to engineer virtual players. As it can be seen in Figure 2-9, the framework is broadly composed of: (i) Data Processing Module (DPM), (ii) Monitoring Service (MS), (iii) ***Jason*** Platform and (iv) Avatar in the virtual world.

In principle, the perception of surroundings in virtual world is the main responsibility of avatars (e.g. NPCs). In addition to this, a Data Processing Module is attached to the whole system in order to improve physical awareness through the complex events detection mechanism which is able to interpret low level sensor data received from virtual worlds by a set of data inference mechanisms. These collectives constructed by simple observation as well as inference are able to deliver higher accuracy in representing the current state of the VW abstractly, and in turn provide richer percepts for the ***Jason***, BDI-type cognitive agents taking a role of reasoning agent, on the ***Jason***

Platform.

The regulation approach is implemented by the combination of this ***Jason*** agent and Monitoring Service where the improved social environment awareness takes place. Actually, the central role of Monitoring Service is to inspect whether social expectations[2] are fulfilled or violatied in the behaviour of a football player. The Monitoring Service is composed of: (i) states representing the type of social situations that provides what kind of behaviours could be correct and (ii) a set of relationship rules identifying the consequences of events resulting from the Data Processing Module and expected behaviour in the social interactions wherein multiple players are engaged. Once the events detected or those inferred by the Data Processing Module have triggered either a goal or a series of actions in the ***Jason*** agent program, the Monitoring Service starts the inspection of fulfillment and violations, consistent with the underlying mechanisms of normative frameworks. After the detection of events an agent brings about, the Monitoring Service performs reasoning over whether events by an agent satisfy the social expectations described in the service or not.

Depending on the monitoring result, the ***Jason*** agent coupled with the virtual football player produces the socially adequate response under the current situation. In here, the result that whether or not an agent comply situationally specific social expectations becomes a new percept in the ***Jason*** agent. This percept is a source to trigger a goal or a series of actions as a belief addition event in the ***Jason*** platform. The decision on which plan to execute next is determined by taking into account the result that whether to comply with the expectation or not, subject to the set of beliefs an agent currently possesses, which influences on the selection condition of a plan.

Listing 2.1 : Social Expectation Example [Ranathunga, 2013]

```
1  @Plan1
2  +successful_pass(OtherAgent, Me)[state(N)] :
3          current_tactic(give_and_go(OtherAgent, Me))
4          & .my_name(Me)
5          & in_situation(opposition_leading)
6          <- .term2string(OtherAgent, OtherAgentStr);
7          .concat("('U',",
8                  "'advanceToGoalB(", OtherAgentStr, ")',",
9                  "'penaltyB(", OtherAgentStr, ")')",
10                 Expectation);
```

[2]According to Mitchell [Mitchell, 1973], the social expectation stands for norms during social interactions

```
11          monitor.start_monitoring("fulf", "move_to_target",
12                  "expectation_monitor",
13                  "#once", Expectation, [N]);
14          monitor.start_monitoring("viol", "move_to_target",
15                  "expectation_monitor",
16                  "#once", Expectation, [N]).
```

Listing 2.1 shows some sample code in ***Jason*** that shows how social expectations influence the virtual football players' behaviour. Let us assume the `Me` agent is successfully passed the ball from `OtherAgent` and all conditions in `Plan1` are satisfied to trigger `Plan1`. According to the plan body, the `Me` agent is expected to fulfill the action `move_to_target` (line 11 – 14). Thus, `monitor` starts to monitoring whether the `Me` agent complies with the `move_to_target` action (line 11) or not (14). As we can see, an event `fulf` will be generated when `move_to_target` is fulfilled, and an event `viol` otherwise.

Listing 2.2 and 2.3 show plans which can be chosen depending on the result of Monitoring Service (Line 1 and 1). When the result of monitoring is detached, `monitor` stops monitoring and the agent `Me` carries out the relevant action `aci_in_vw` (Line 10) or `choose_and_enact_new_tactic` (Line 9) in response to `+fulf` or `+viol`, respectively.

Listing 2.2 : Plan for Fulfillment [Ranathunga, 2013]

```
1  @PlanForFulfillment
2  +fulf("fulf_reach_penaltyB")
3          <- monitor.stop_monitoring("fulf",
4                  "reach_penaltyB",
5                  "expectation_monitor");
6          monitor.stop_monitoring("viol",
7                  "reach_penaltyB",
8                  "expectation_monitor");
9          //turn towards other agent and pass the ball to him
10         act_in_vw(``pass","up-kick").
```

Listing 2.3 : Plan for Violation [Ranathunga, 2013]

```
1  @PlanForViolation
2  +viol("viol_reach_penaltyB")
3          <- .stop_monitoring("fulf",
4                  "reach_penaltyB",
5                  "expectation_monitor");
6          .stop_monitoring("viol",
7                  "reach_penaltyB",
```

```
8                   "expectation_monitor");
9           !choose_and_enact_new_tactic.
```

One important aspect in these two examples (Listing 2.2 and 2.3) is that the violation is allowed for virtual football player, in contrast to the previous approaches (e.g. reflex agents with norms, utility based agents and regimentation approach). This reminds us that the monitoring and its result which represent a normative framework are not able to enforce the direct control of football player behaviour. In other words, the virtual football player in VW is free from the obligation that norms (or social expectation) should be complied with any time, any place. Instead, the football player can choose the actions with respect to its own mental states in ***Jason*** agent, which is constructed by both the result of expectation monitoring service and individual perception mechanism incorporating with Data Processing Module, an inference based event detector.

In line with this research, Baines *et al.* [Baines and Padget, 2014] propose a simulation framework inspired by the regulation approach. The main objective of the framework is to design and simulate virtual vehicles behaviour controlled by ***Jason***, a (BDI-type) cognitive agent, under the governance of the normative framework. The work is motivated by the fact that the driving behaviour is able to be heavily influenced not only by individual preference (e.g. fuel consumption, CO_2 emmissions) but also by environmental collectives in traffic (e.g. interactions between vehicles, traffic flows, road conditions). Thus, this framework and its simulation pursue the demonstration of the impact (whether positive or not) of situational awareness accomplished by normative frameworks on the determination of individual virtual vehicles' driving patterns.

The system overview is shown in Figure 2-10. The simulation framework is composed of (i) normative framework, (ii) ***Jason*** agent and (iii) virtual vehicles. The main role of each component is almost same as those in the previous work. Virtual vehicles take a reponsitility to perceive external events, to distribute its percepts to both normative framework and ***Jason*** reasoning agent, and to embody a series of actions when ***Jason*** agent choose a specific plan to execute. With respect to virtual vehicles, the normative framework performs the social reasoning about context by means of the Institutional Model [Cliffe, 2007], detaches a new set of norms agent, and in turn broadcasts this to the ***Jason*** agents. In the mean time, the ***Jason*** agent carries out the individual reasoning with ordinary percepts as well as the set of norms to find the most appropriate plan in the current situation.

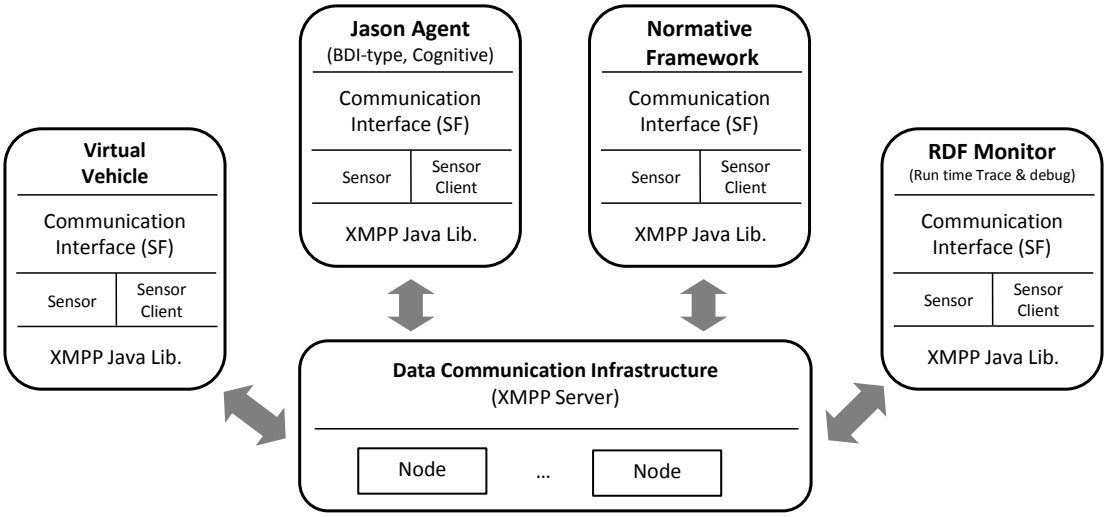

Figure 2-10: *Simulation Framework for Norm-Governed Virtual Vehicles. (c) Baines, 2014. [Baines and Padget, 2014]*

The set of norms could be events to trigger a certain goal in the agent program or percepts to update beliefs in the agent mind. In doing so, the ***Jason*** agent may be able to have more chances in the selection of plans, between norm-triggered and individual contextual information triggered goals. This selection is achieved by the practical reasoning process in the ***Jason*** agents. In this sense, the use of norms and its associated normative framework in agent behaviour can be regarded as a means to provide complemenatry information identifying the appropriate patterns of behaviour with regards to the current situations. Since these norms initially originated from the collectives of multiple vehicles (which can be seen as multi-agents), norms might be able to present improved understanding of the social situation for individual agents.

In conclusion, the works inspired by regulation approach presented here have contributed to not only the advances of virtual characters autonomy on norms and goals, but also support norm compliance at the level of individual agents. In general, norms can be a kind of recommendation/guidance for virtual characters, but cannot control the agent behaviour in strict obedience to norms by using reasoning agents in conjunction with characters.

However, as we saw in two examples, in conventional BDI agents, norm compliance is typically achieved by design. That is by specifying plans that are triggered by detached norms, because the agent programmer knows which norms the agent will adopt, and then prioritising those rules so that those supporting norms are chosen over those

preferred by the agent's mental attitudes, in order to suppress conflicts between the normative and the agent's existing goals. In addition, when an agent encounters new and unknown norms, which were not taken into account at design time, there is typically no plan to deal with those norms in the plan library at run-time. Hence, norm compliant behaviour cannot normally be exhibited because the norms are unavoidably ignored. Yet worse, agents may suffer a punishment from the enforcement of the normative system as a result of a violation caused by their incapacity to process the normative event. However, a solution that applies a hierarchical prioritisation of normative over ordinary plans would still deprive an agent of its autonomy, since the norms in effect are treated as hard constraints, whose violation is not possible.

2.4 Discussion

We summarise the system features and capabilities in engineering norm-aware virtual characters in Table 2.2. As noted in Section 2.1, the systems are categorised by: (i) the use of the infrastuctures (i.e. normative frameworks) where reasoning about norms takes place, and (ii) the type of agents which carry out norm-aware individual reasoning (i.e. a deliberation on norms, personal goals and sanctions).

In case without infrastuctures, where norms are internal knowledge in a single agent, the straightforwardness in modelling and building normative behaviour in virtual characters is a strong advantage. However, it is obvious that this approach reveals a lack of flexibility in social norms subject to changing of situations. Since social norms in an agent mind are likely to be pre-defined (or pre-designed) already in relation to the specific situations at the design time, these norms are too domain-specific to be adaptive to the frequent changes in virtual environments. When the situation is changed, those social norms might no longer be valid, thus less adequate or even worse totally wrong in that situation. The other drawback is that the system always requires explicit action to evoke normative behaviour both in the reactive plans (in Human Territoriality Model [Pedica and Högni Vilhjélmsson, 2010]) and in the process of goal selection (in Thespian [Si et al., 2006]). However, external events which actually occurred in some place but are not observed by the virtual character might bring about new situations requiring normative behaviour in the virtual environment. In this instance, a single agent approach is not able to exhibit appropriate (normative) behaviour due to limited

	Norms Infrastuctures			Individual Agent		
	Formal Model of Norms	Design Norms	Reasoning Norms	Agent Type	Violation Norms	Norm-Aware Reasoning
Human Territoriality [Pedica and Högni Vilhjélmsson, 2010]	No	No	No	Reflex	No	No
Thespian [Si et al., 2006]	No	No	No	Utility Based	No	No
Virtual Institution [Bogdanovych, 2007]	Yes	Yes	Yes	N/A	No	No
v-mWater [Almajano et al., 2013]	Yes	Yes	Yes	N/A	No	No
Expectation Model [Ranathunga, 2013]	Yes	Yes	Yes	Goal Oriented	Yes	No
Virtual Vehicles [Baines and Padget, 2014]	Yes	Yes	Yes	Goal Oriented	Yes	No
DNA^3	Yes	Yes	Yes	Goal Oriented	Yes	Yes

Table 2.2: *Comparisons of Systems for Norm-Aware Virtual Characters*

comprehension of the current situation, which is a consequence of not perceiving an explicit event that triggers a reactive plan or a goal selection.

In contrast, the approach using infrastructures promises better comprehension of the situation due to the capability in reasoning about norms depending on changes in virtual environments. The infrastructures observe the occurrence of a series of external events produced by multiple virtual agents, so as to bring about situationally adaptive norms in response to changes of situation in the environments. Therefore, virtual characters are able to exhibit desirable behaviour in the new situation with the consideration of a guidance that the infrastuctures offer.

However, the level of norm-autonomy of agents is limited in this approach. Indeed, the Virtual Institution [Bogdanovych, 2007] does not use the intelligent agent to determine whether to comply with norms or not. Instead, virtual characters simply comply with a set of norms the normative framework (Electronic Institution in this case) detaches without further reasoning on that. The other two, Expectation Model [Ranathunga, 2013] and Virtual Vehicles [Baines and Padget, 2014], are in the better situation where goal-oriented agents perform norm-aware reasoning. But the reasoning on norms and goals is relatively simple, not unlike that of both reflex and utility based agents. In effect, norms have its own properties such as 'when' (e.g. deadline), 'what' (e.g. required behaviour), deontic forces on the behaviour (e.g. permission, obligation, prohibition) and sanctions when the required behaviour is violated. The conventional goal-oriented agents (e.g. BDI type agents) are not capable of dealing with those properties such as assessing the essential properties between norms, goals, prohibitions and sanctions. We believe that norm-aware reasoning can promise the choice of the most appropriate behaviour, based upon a preference of individual agents or a level of autonomy, as allowed by the infrastructure.

Against this background, we propose **DNA**^{3} which pursues a hybrid approach to take advantage of both approaches to engineer norm-aware virtual character. As seen in Table 2.1, **DNA**^{3} facilitates the 'infrastructure', which is capable of specifying, reasoning about norms, for the purpose of a better comprehension of situations, which in turn provides situationally appropriate norms subject to frequent changes in the environments. In the meantime, 'norm-aware reasoning agent' is able to deliberate on norms, goals and sanctions is coupled in the system so that virtual characters can organise the choice of behaviour, taking into account norms and individual context.

2.5 Summary

In this chapter, we present a state-of-the-art survey of the domain of virtual character behaviour incorporating norms and normative frameworks. In brief, we investigate two main approaches: (i) norms as internal, pre-defined knowledge in a single virtual agent mind, and (ii) norms as external knowledge in the virtual environment which can specify and reason about norms and thus broadcast new sets of norms to individual virtual agents.

In more detail, we introduce the use of both reflex agents (Human Territoriality Model [Pedica and Högni Vilhjélmsson, 2010]) and utility based agents (Thespian [Si et al., 2006]) for reasoning about the normative behaviour of virtual characters in the first approach. Afterwards, the use of normative systems to guide individual agents behaviour is presented, which can be decomposed into (i) the deployment of an Electronic Institution into the virtual environments (Virtual Institition [Bogdanovych, 2007], *v-mWater* [Almajano et al., 2013]), (ii) the combination of expectation monitoring service and goal-oriented agents (e.g. BDI type cognitive agents) (Expectation Model [Ranathunga, 2013]) and (iii) the utilisation of the institutional model and goal-oriented agents (Virtual Vehicle [Baines and Padget, 2014]). A comparison of systems with regards to features and capabilities in building norm-aware virtual characters is shown in Table 2.2 with a discussion of advantages and disadvantages of each system. At the end of the discussion in Section 2.4, a hybrid approach which combines normative frameworks and norm-aware individual agents is proposed as a way of improving the norm-awareness in virtual characters behaviour.

In practice, infrastructure for NorMAS has been developed but a few integrations into the virtual environment has been proposed. This is mainly caused by heterogeneity such as differences in programming languages to develop the softwares and platforms (or OS) to run softwares, which hinders the integration of software components (e.g. intelligent agents, virtual characters and normative frameworks) as discussed in Section 3.1. As a result, virtual environments that allow AI programming for virtual characters take such a tightly-coupled approach that addition or removal of components is not straightforward [Adobbati et al., 2001, Bogdanovych et al., 2008c, Gemrot et al., 2009, Ranathunga et al., 2011, van Oijen et al., 2012, Veksler, 2009]. In the next chapter (Chapter 3), we introduce an integration middleware, Bath-Sensor-Framework, its engineering methodology and illustrative examples which show how intelligents agents,

virtual chracters and additional software components can be easily connected to this end.

Chapter 3

A Middleware Approach to Connect Cognitive Agents and Virtual Environments

As discussed in Chapter 1, norm-aware virtual characters are the primary subject of investigation in this thesis, in order to design and simulate human-like (social) behaviour of virtual characters. For this purpose, the research centres on characters reasoning about their behaviour with (social) norms through the *socio-cognitive* approach [Stein, 1993, Bandura, 2001, Akgün et al., 2003] as proposed in Chapter 1, which emphasises the incorporation of individual cognitive reasoning and social reasoning on human choices, in particular when individuals are engaged in social settings.

As a starting point to this end, we introduce a middleware approach to support the coupling of (BDI-type) cognitive agents and virtual environments and their participants as noted in the Chapter 1. It enables the control of virtual characters' behaviour through cognitive reasoning, as humans do in decision making for their individual autonomy.

We firstly propose an integration middleware, Bath-Sensor-Framework (BSF) with an investigation including the architecture, main functionalities (e.g. sensors, pub/sub mechanism) and programming model. Given BSF, we subsequently show an illustrative example of the integration between ***N-Jason*** (a BDI-type norm-aware cognitive agents as an intelligent agent. See Chapter 5 for more details) and *SecondLife* [Linden Labs, 2014] (a virtual environments where virtual characters reside). In this case study, internal operations such as data and control flows between the components are presented in

detail.

3.1 Introduction

Programming the environment in which a multiagent system is situated has been and continues to be an active research issue [Ricci et al., 2011]. From this perspective, many rich open systems, formed from networked 3D Virtual Environments (VEs) such as online games, non-gaming applications or some entertainment context are all potential (programmable) environments, since they offer sufficient variety to simulate real and semi-real world situations. *SecondLife* [Linden Labs, 2014] is an obvious representative example of a 3D virtual environment: it provides a sophisticated, dynamic and realistic virtual world as if duplicating the modern human society with avatars and 3D objects [Kumar et al., 2008]. Such a virtual world may encourage advances in agent intelligence, through the demands of sensing and interaction, as agents are situated in increasingly complex, dynamic or realistic environments.

However, the integration of agent software and rich environments creates a range of challenges, arising not least from the variety of each and that neither is typically designed to interact with the other. For example, the original purpose of *SecondLife* is to provide an avatar in a networked 3D virtual environment, that it is expected a human will control, so it does not explicitly take into account either the use of AI or integration with other applications. The ***N-Jason*** agent platform proposed in Chapter 5 is similarly placed: its objective is to provide a BDI-based a deliberative reasoning engine for agent research, so it does not consider standard programming interfaces for other environments.

As a result, research on agent-environment programming has mostly relied on tightly coupled approaches, characterised by using a specific ontology, protocol and interface that are particular to one system [Adobbati et al., 2001, Trescak et al., 2011, Gemrot et al., 2009, Ranathunga et al., 2011, van Oijen et al., 2012, Veksler, 2009]. Such a lack of interoperability is possibly not beneficial overall for the development of agent intelligence because there is little scope for the agent that is built for one VE to be exercised in another and so the agent is unavoidably mono-cultural. Besides, such tight connections between agents and particular environments must somehow inhibit further potential applications arising from alternative agent-environment combinations. We

believe that a way forward from this situation may be possible through an appropriate form of middleware.

Thus, our objective is to describe and to demonstrate a kind of integration middleware that serves not only to loosen the coupling between agents and environments, but also to make it possible to consider the connection of any kind of agent and any kind of environment. In software engineering pattern terms, we outline a façade for each agent platform and each environment, where each communicates with the other by means of events (in effect, asynchronous message passing), facilitated by the use of a publish-subscribe server. This constitutes the essence of the Bath Sensor Framework (BSF), which provides the means to link software components independently of programming language, platforms or operating systems, so in principle offering good accessibility, distribution and scalability as an agent-environment integration framework. Performance is a more delicate issue that will take time and experience to establish, depending on the communications overhead (the pub/sub server) and – more likely to dominate – on the decision-making cycle of the agent, although this will clearly depend on the sophistication of the agent architecture.

The remainder of this chapter is organised as follows. The overall system design is be described in section 3.2 including the introduction to the integration middleware, BSF. Section 3.3 presents the illustrative example and and in depth description of internal operations in the coupling on BSF. This section also includes the evaluation of the system. We finish with a brief survey of related work in section 3.4 followed by summary in section 3.5.

3.2 System Design

In this section, we describe the system design and how it can integrate an agent platform with a virtual environment. In particular, we will demonstrate the interaction between the software components and describe the programming model.

For our experimental set-up, the collection, distribution and exchange of data is performed by using publish/subscribe between event producers and consumers via the Extensible Messaging and Presence Protocol (XMPP), an open standard communications protocol [XMPP Standards Foundation, 0129a]. Although XMPP is often cited as a component in real-time (web) systems [XEP-301: In-Band Real Time Text, 2012],

there is little quantitative evidence to back this up. In consequence, we have carried out some preliminary evaluation (see 3.3.2) and we return to this issue in related work. For an agent platform, we use ***N-Jason***, a BDI type cognitive architecture and as VE, we use *SecondLife*.

Any of these components (***N-Jason***, SL, XMPP) may be substituted in pursuit of a better fit with the requirements set out earlier, but the primary focus of this work is our evaluation of the adequacy of the BSF, instantiated as outlined, as a solution to the issues identified above pertaining to agent-environment programming. In doing so, we present an illustrative example that connect ***N-Jason*** agents and *SecondLife*, and discuss how to accommodate differences in platform and programming language.

3.2.1 Overall System Architecture

The essence of the system architecture is that the agent platform is decoupled from the virtual environment by means of a publish-subscribe messaging server – in this case, XMPP – as shown in Figure 3-1.

In the case of *SecondLife*, the virtual character is created using the OpenMetaverse library (LIBOMV [OpenMetaverse Organization, 2012]), which also provides the connection to the *SecondLife* server. The role of the virtual agent is to interpret the actions received from the BDI agent, and then carry out the resulting "physical" actions. In the other direction, the virtual character perceives the environment and the percepts are delivered to the BDI agent via BSF, where they become a belief that influences the agent's reasoning process.

Clearly the BSF plays a key role in facilitating the interaction between the two components. In particular, through the imposition of a simple communication API, a java-based agent platform and a virtual character, in this case written in a different language and running on a different platform, can interact with one another. We now explain in more detail about the sensor framework.

3.2.2 Bath Sensor Framework

The Bath Sensor Framework (BSF) is an abstraction layer for data collection, distribution and exchange built upon XMPP technology. The primary task for which the framework was conceived is the effective collection of data from numerous physical or

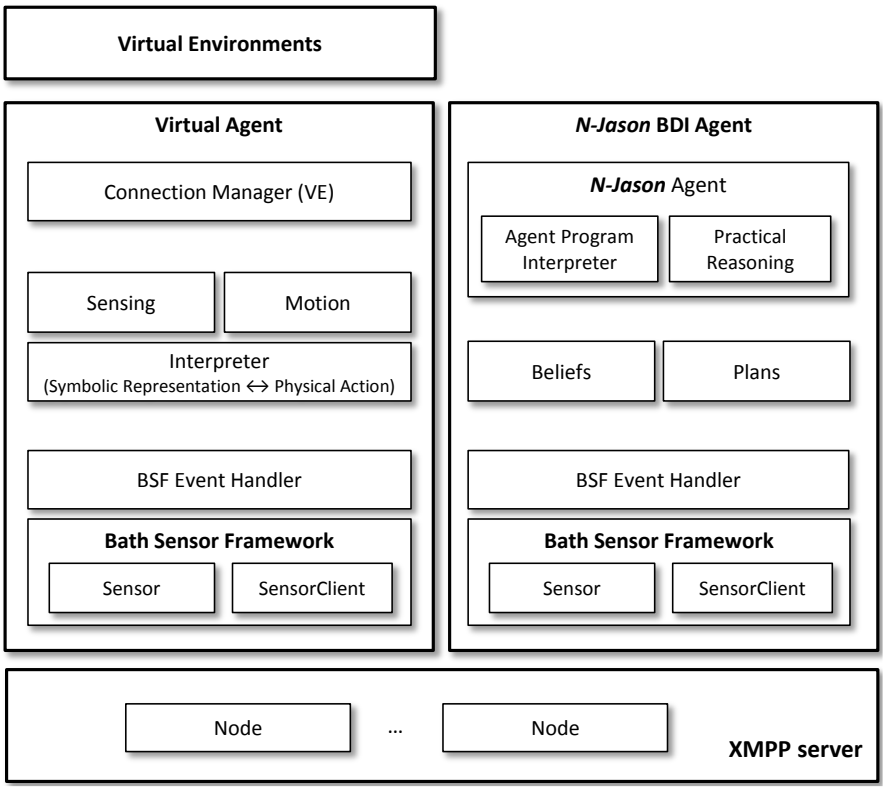

Figure 3-1: *System Architecture of an Example*

logical sensors, and its subsequent distribution to the relevant devices or software connected to the XMPP server. The data itself is represented in RDF or JSON, although this is not mandatory: the XMPP message structure is just a HTTP body and can be any representation that is suitable.

XMPP is an open standard communication protocol built upon a set of open XML technologies [XMPP Standards Foundation, 0129a]. It is intended to provide not only presence and real-time communication services, but also interoperability by exchanging any type of data in cross domain environments by means of nodes in the XMPP server. To this end, it supports 1-to-1, 1-to-many, and many-to-many data transport mechanisms, so that any data may be be transferred from anywhere to anywhere [Bernstein and Vij, 2010b]. Its flexibility, performance and lightweight nature have lead to XMPP being chosen to support research in a diverse range of fields, including Many Task Computing [Stout et al., 2009], bio-informatics [Wagener et al., 2009] and Cloud Computing [Bernstein and Vij, 2010a], as a data distribution service in preference to HTTP or SOAP services.

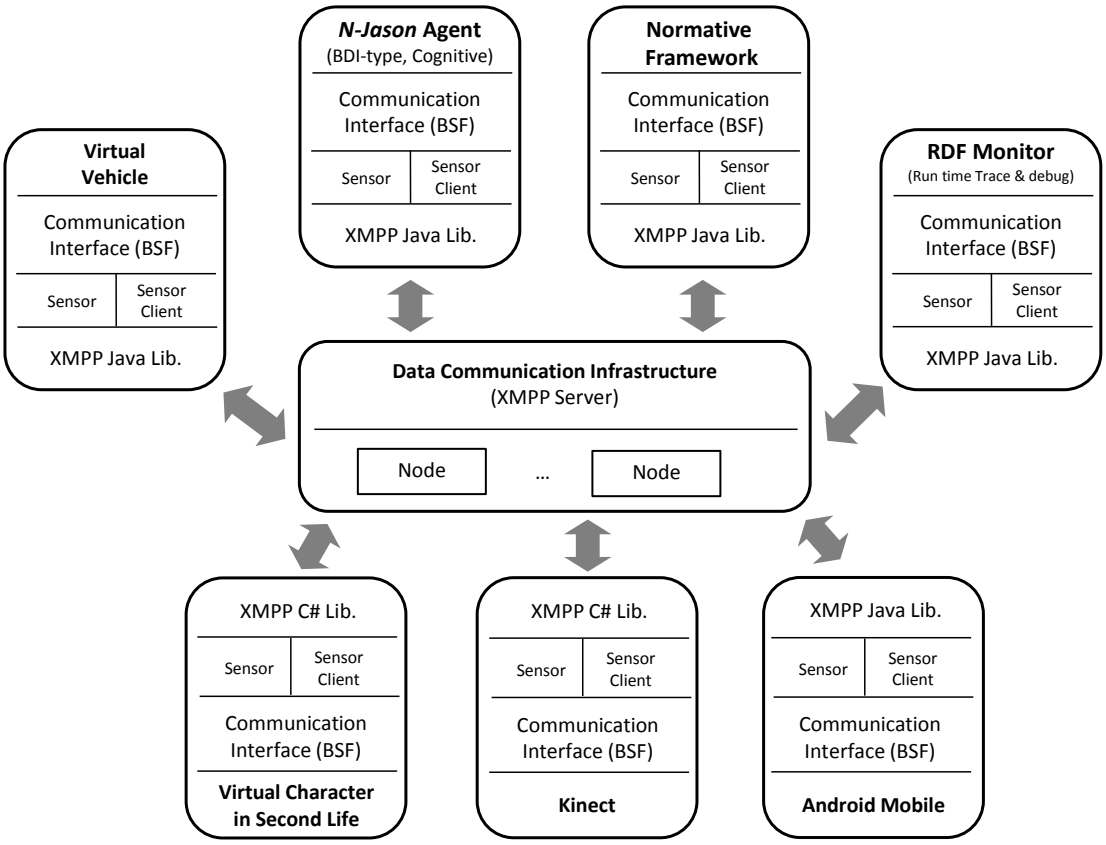

Figure 3-2: *The System Architecture of Bath Sensor Framework (BSF)*

The above features suggest a number of advantages over pure TCP/IP connections. The latter typically require quite careful set-up and can be fragile where the connection graph is not simple. Moreover, TCP/IP is primarily for 1-to-1 connections, so every additional connection needs an independent additional socket whenever multiple software components are integrated into the main system. In contrast, XMPP provides a star- or bus-like connection model to resolve the m-to-n problem, but through the node abstraction within the server, allows the set-up of multiple virtual circuits. Furthermore, the data producer does not need to know the consumer's identity to set up the connection and through the server's mediation of the connection, the system acquires a degree of fault-tolerance, and permits the observation of system behaviour by third parties, rather than having to replicate such mechanisms in each component. Thus, we conclude that XMPP offers several attractive features, which is why we have chosen to base BSF upon it.

For our purposes, the most notable feature of XMPP and hence the BSF is its pro-

vision of a publish/subscribe mechanism. BSF supports these operations by means of *Sensor* and *SensorClient* classes. When the *Sensor* is created in the application which has a role as a data source, a corresponding node is also created in the XMPP server[1]. Once created, the application can publish the data sensed from the real or virtual world via the node. If the *SensorClient*, which is created in another application, and has a role as a data consumer, sets up a subscription request on that node and registers a corresponding handler in its application, then the data will be transferred from one application to the other. As noted earlier, the data is represented in RDF or JSON and published data can be stored in a triple-store (in our case OpenRDF) so that historical data can be retrieved on request from the *SensorClient* using the SPARQL query language[2]. In this way, BSF supports a form of messaging passing with unstructured data between multiple software components.

A particularly valuable aspect of XMPP/BSF is its relative independence from both operating systems and programming languages so that more general programming environments can be provided for users attempting to combine heterogeneous software components. Thus, regardless of language, its interfaces reveal the same classes, methods and data structures, so that all kinds of applications or libraries can be integrated relatively easily just by adding the classes inside applications.

As can be seen, the features of this framework present a simple and flexible programming environment for heterogeneous software components, with a good level of a accessibility in terms of a simplicity of protocol and ease of connection, performance, and distribution. Consequently, we show how the BSF can facilitate the integration of a BDI type cognitive architecture for virtual agents. The next section discusses the programming model of the BSF.

[1]A node (also called topic) is a focal point for the publication and subcription, to which the publisher (e.g. *Sensor*) sends data, and from which subscriber (e.g. *SensorClient*) receives the notification. The node is a topic-based: when xmpp entities (e.g.*Sensor*) are created to share information about specific subject (e.g. percepts), the corresponding nodes are also created. The pub/sub service at nodes is grounded upon "*Observer*" design pattern: when a *Sensor* publishes information at the node, an event notification is broadcasted to all *Sensorclient*s which subscribe to that node [XMPP Standards Foundation, 0129b].

[2]Other (structured, relational) databases may equally be connected to the Openfire [Ignite Realtime, 0129b] XMPP server and accessed by SQL queries.

Listing 3.1 : Example of Sensor object

```
1  public class SampleVirtualAgent extends Sensor {
2    public SampleVirtualAgent(String server,
3           String user, String pwd, String node) {
4        super(server, user, pwd, node);
5      }
6
7    public void run() {
8      while (true) {
9        if (usePub) {
10         DataReading data = new DataReading();
11         publish(data);  // populate
12       } else if (useMsg) {
13         sendMessage(message);
14       }
15     }
16   }
17 }
```

3.2.3 Programming model of Bath Sensor Framework

The Bath Sensor Framework can equally be applied to data exchange between distributed software components as to sensor based applications. The perspective of this section is limited to the former. The analogy we draw, in making the connection between BDI agent, virtual agent and virtual environments, is that the virtual agent can be viewed as a sensor for the percepts from the environment, in line with the traditional view of the "situated" (intelligent) agent [Wooldridge, 2009]. In this context, the *sensor* object is instantiated in a virtual agent in order to collect percepts for the BDI agent. Conversely, the BDI agent needs a *subscriber* object to receive the percepts and subsequently reason over acquired beliefs.

To publish data from the data source to the target, subclassing via extending *Sensor* class is necessary inside the virtual agent (see Listing 3.1). In a data consumer such as in the BDI agent, the *SensorClient* object has to be instantiated inside the BDI agent (see Listing 3.2). The C# version of this example is identical modulo the grammar of the programming language.

The objective of the design is that it should suffice just to put the *Sensor* and *SensorClient* object in a wrapper around whichever software component it is desired to integrate into the event processing framework.

Listing 3.2 : Example of SensorClient object

```
1  public class SampleAgent {
2    public static void main(String[] args) throw Exception {
3      SensorClient sc = new SensorClient(server, user, pwd);
4      sc.addHandler(nodename, new ReadingHandler() {
5        public void handleIncoming(String node, String rdf) {
6          DataReading dr = DataReading.fromRDF(rdf);
7          // doing something
8        }});
9      sc.subscribe(nodename);
10   }
11 }
```

3.3 Illustrative Example

The aim of this section is to demonstrate how the integration of agents and VEs is enabled by BSF. In particular, this example focuses on the impact of the use of BSF on the integration of agent platform and virtual environment, in respect of some desiderata for computational models such as generality, modularity and dynamic extensibility [Ricci et al., 2011]. A complementary aspect is the increased capacity for distribution of the components of the software architecture, so that it is not so tightly coupled and that the addition or removal of components is straightforward.

In the preceding section, we outlined how *Sensor* and *SensorClient* objects are incorporated into a BDI agent and a virtual agent. For the following discussions, the components are (i) ***N-Jason***, providing a BDI type cognitive agent, (ii) the OpenMetaverse library, providing a virtual agent, and (iii) *SecondLife*, providing a virtual environment all linked by the BSF.

3.3.1 *N-Jason* agent and *SecondLife*

The goals of this work are two-fold: (i) to demonstrate the integration of ***N-Jason*** agents with avatars in *SecondLife*via BSF and (ii) to identify appropriate mechanism for the control of avatars via BSF, through exploratory scenarios.

The brief scenario for this section is as follows: one avatar controlled by a human in *SecondLife* server says 'hello' to a *SecondLife* avatar governed by a ***N-Jason*** agent. In what follows, we refer to the ***N-Jason*** controlled avatar as the *SecondLife* Bot (SLB). When the SLB receives the greeting message, the SLB sends it to the ***Jason*** agent over

XMPP via the *Sensor*, where it is received via the *SensorClient*. The ***N-Jason*** agent then updates the percepts, and performs one cycle of reasoning. As a result, the belief 'hello' triggers the plan 'bow', and appropriate actions are sent to the SLB. Finally the bot does a 'bow' animation by means of the OpenMetaverse library after interpreting the action plan 'bow'. Interpretation of the action plans means the conversion of actions from ***N-Jason*** to *SecondLife* animation action(s). For example, if 'bow' is received from ***N-Jason***, then SLB looks to see whether 'bow' is defined in the action map: if so it perform that animation. More commonly, an action plan is likely to be composed of several atomic actions (or animations) in SLB.

A notable aspect of this example is that two heterogeneous software components are able to interact by means of the BSF: because the OpenMetaverse library is in C#, so too is the SLB, but ***N-Jason*** is Java. Previous work has been able to integrate them by means of the .NET framework [Ranathunga et al., 2011], but this requires all the components to be in the same location, on a specific platform and also couples them quite tightly. The C# interface to BSF is achieved by an extension of the jabber.net library [Jabber-Net,], whilst the Java interface is built on the Smack library [Ignite Realtime, 0129a], although this is just one of several available Java libraries for XMPP. We are currently using the OpenFire [Ignite Realtime, 0129b] XMPP server, although again there are several other candidates.

N-Jason agent and Bath Sensor Framework

Figure 3-3 sketches the basic operations between a ***N-Jason*** agent and the BSF. The ***N-Jason*** agent is extended, using the `AgArch class`, with the *Sensor* and *SensorClient* objects. Percepts from the SLB are received by the *SensorClient* object, which results in updates to the beliefs. The next reasoning cycle utilises these beliefs to retrieve an action plan and the *Sensor* object publishes the plan to the SLB.

SecondLife Bot and Bath Sensor Framework

The OpenMetaverse library[3] [OpenMetaverse Organization, 2012] provides a set of APIs to program a Second Life avatar in terms of creation, appearance, movement, communication – verbal and non-verbal – and interaction with each other, in the same

[3]The OpenMetaverse library is available via `https://github.com/openmetaversefoundation`

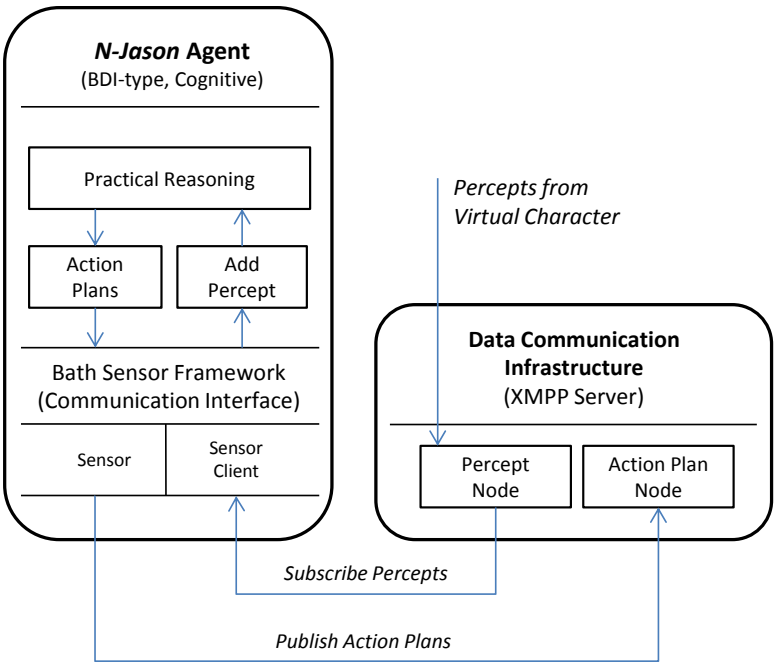

Figure 3-3: *Basic Operation between* **N-Jason** *agent and Bath Sensor Framework*

way as the Second Life official viewer application. Thus, with OpenMetaverse, it is possible to program complex compound actions in the avatar.

Figure 3-4 shows the basic operations between *SecondLife* bot and the BSF. Once logged in, the SLB appears as an avatar in *SecondLife*Às such, it has an identity and can move and interact with other participants, as well as perceive events taking place nearby the SLB itself. Consequently, all events occurring in *SecondLife* are detectable in the OpenMetaverse library and delivered via a callback mechanism to the *Sensor* object in the SLB, which collects them and publishes them for the ***N-Jason*** agent to receive. On the other side, the *SensorClient* object receives (subscribes to) the action plans from the ***N-Jason*** agent. These are then translated into sequences of atomic actions, which are a combination of defined actions in OpenMetaverse or user-defined actions. As a result, the SLB carries out these actions in respect of other participants or its environment.

3.3.2 Evaluation

We have prototyped a demonstrator using the BSF: ***N-Jason*** agents controlling *SecondLife* avatars in a *SecondLife* virtual environment. At the outset, our informal requirements were stated as (i) accessibility: connection of new software components

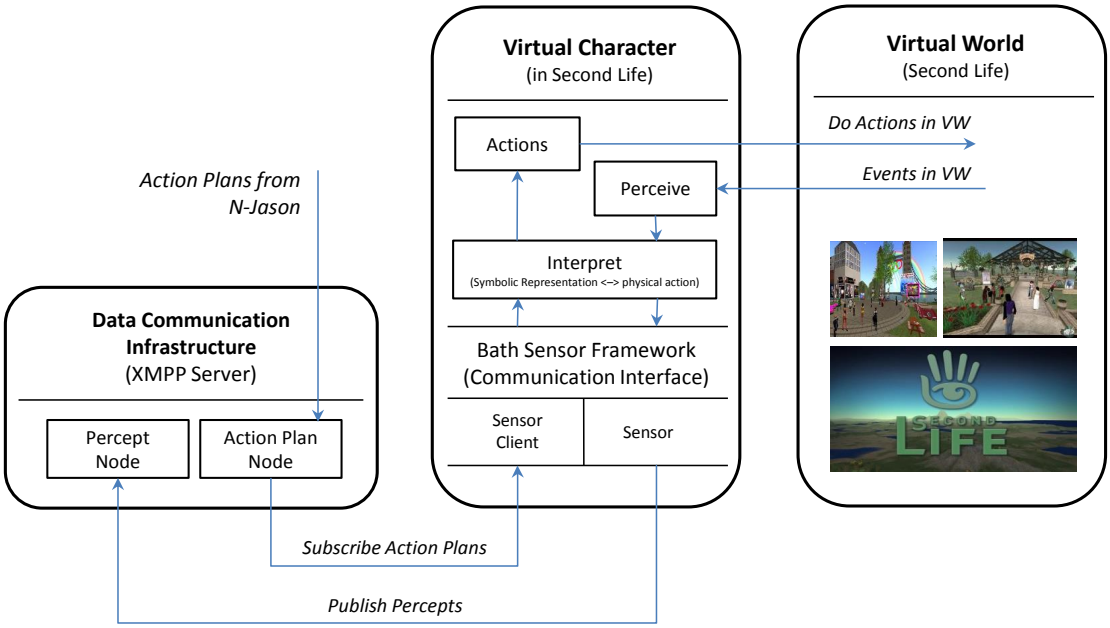

Figure 3-4: *Basic Operation between SecondLife Bot and Bath Sensor Framework*

(ii) performance: decoupling without significant degradation (iii) distribution: connection of components where-ever they might be, and (iv) scalability: in terms of size of environment and number of participants.

Clearly, at this stage, progress on scalability is not feasible, but we can comment on each of the other aspects, although we devote the most space to performance because seems to be the one that raises the most questions.

Accessibility

Whilst each case study started out with the decoupling of the decision-making components (BDI agent) from the virtual environment, each has added other components that demonstrate the practice of our accessibility requirement.

In the context of the case study, connection with the VE presented a challenge, because the OpenMetaverse library is written in C# and runs in .NET, hence required the construction of a C# client for the BSF, as well as cross-platform communications. Subsequently, we have incorporated a connection with an institutional model (involving Java and Answer Set Programming), following the initial implementation of Balke *et al.* [Balke et al., 2011], but decoupled by means of the BSF interface (which is pre-

sented in Chapter 4). This has been used to demonstrate norm-mediated behaviour of agents in *SecondLife* (the 'hello' example described earlier, and a more complicated one involving making space in a queue for an individual who is given priority)[4].

Performance

We have carried out preliminary performance evaluation of BSF by measuring the elapsed time during the publication and subscription, which may reflect the physical layer latency on the network. In particular, we focus on the investigation of the time just before the publication of data that is already packed as a single item, and taking the time when the subscription handler detects the arrival of the item. Two cases are explored between (i) Java entities, e.g. BDI agent–virtual worlds (developed in Java) and (ii) Java and C# entities, e.g. the virtual agent–BDI agent or vice versa. The hardware platform used for this evaluation is a Windows 7 Professional edition, comprising intel Core i7 Quad 64 2.93 GHz each with 4MB cache, with 6GB RAM. The hardware platform for running the XMPP server is a Windows 7 Starter edition, comprising intel Atom CPU N455 32 1.66GHz with 1GB RAM. Both hardware systems are connected to a (VLAN) network implemented on (copper) Gigabit ethernet.[5]

Elapsed Time				
	Published every $25ms$		Published every $100ms$	
	μ	σ	μ	σ
Java to Java	1.875 ms	0.625	-	-
C# to Java	1.119 ms	0.630	-	-
Java to C#	2.826 ms	7.557	1.156 ms	2.268
C# to C#	2.664 ms	7.829	1.150 ms	2.466

Table 3.1: *Delivery Time of BSF*

During each evaluation, 152139 items (2 elements, metrics and commands each) are published every 25msec. During the whole evaluaions, no data losses are observed on the subscriber side. The overall statistics are shown in Table 3.1, where μ and σ are the mean and standard deviation, respectively. These results suggest that transport latency seems acceptably low. Based upon the largest average elapsed time (2.826ms),

[4]For more details, see Chapter 4

[5]The full test set is available via `https://code.google.com/p/bsf/downloads/list`, retrieved 20150130

this implies a frame update rate of at least 350 fps (which stands for 350 messages are delivered per second) in theory, not accounting for time spent on either plan processing or scene rendering. In practice, we observe that subscription in Java applications seems both faster and more reliable, in the sense that the standard deviation is much lower than for C#. C# also appears to be more sensitive to the publication rate, since if the interval is increased (from 25ms to 100ms), the average time falls slightly and the standard deviation improves significantly. We emphasize that performance *per se* is not the objective here, but poor performance or irregularities may trigger problems at higher levels and so it is important to establish a baseline figure for message transmission on a given platform.

Although we have provided statistics on the elapsed time as a preliminary, a qualitative measure of performance might be that the system is performing properly only as long as no events are dropped. That would require even extreme situations to be withing the performance envelope. Quantitative evaluations may not be particularly helpful except perhaps to provide reasssurance that throughput of particular components is probably sufficient not to be the cause of a bottleneck. Even then, it may be hard to say, even if many such components are performing "well", whether their collective performance is adequate. We have, for example, measured elapsed message times between Second Life and the agent controlling an avatar shown in Table 3.1, but this may well say more about the networking infrastructure than about the architecture as whole. As a consequence, it is common to eschew distribution for tight coupling in order to be able to deliver performance guarantees. Thus, performance is less about raw processing power, however that might be measured, but whether the architecture as a whole performs believably. Even then, exhaustive testing all possible states of all possible components is likely to be infeasible, so we are limited, as a poor substitute, to stress-testing individual components as a way of seeking confidence in the overall architecture. In practice, performance is a pervasive issue and tight coupling of components is one way that some control can be exerted over the collective factors that influence it, but in the long term such coupling impacts scalability.

Among the primary (new) sources of delay are the network, which is relatively hard to control, and the XMPP server (or servers, since they may be federated). There are several XMPP server implementations, but it is nor surprising, given the application domain, that all aim for high performance within themselves. We have chosen to use Openfire, because of its stated aims of supporting real time communications projects.

We have not done a comparative evaluation against other XMPP servers. Openfire claims to be able to support significant numbers of (human) users with relatively few resources (e.g up 500 concurrent users with a minimum of 384Mb RAM on a 1.5GHz processor and up to 100,000 with a minimum of 2.0Gb RAM, 2×3GHz processors and 1–4 connection managers). Further details at [Jive Software,] measure factors such as the number of concurrent sessions and packet counts.

Since the fundamental mode of communication is publish/subscribe, one approach to evaluating the processing capacity of a component is to quantify the rate at which it can process incoming items, that is the data in the streams to which it is subscribed. Different components will have different subscription capacities, and depending on their role and where they are connected into the subscription network, one of several approaches may be appropriate if this capacity is insufficient, such as: (i) increasing component input capacity (ii) component replication (iii) throttling input volume, and (iv) inserting an aggregator component whose subscription and publication rates match upstream and downstream components.

There are two forms of mitigation that are possible in the architecture we have outlined: (i) short-circuiting, and (ii) aggregation (as mentioned earlier). Short circuiting is, in effect, taking events from the virtual environment, intercepting them before they are forwarded to the controlling agent, and making a decision that is returned to the VE. We do this, for example, in the Mindstorms scenario, where an android handset is physically located on the robot, which may make some (reactive) control decisions and relay them back over the local bluetooth connection to the (lower level) Mindstorms controller. Several such (nested) feedback loops [Roy, 2007] can be inserted into the control chain depending on need. Such a design pattern reflects a hierarchical control framework, where proximity to source implies lower level events and tighter control, as seen in historical multi-layer agent architectures such as InteRRaP [Müller, 1996] and Touring Machines [Ferguson, 1992]. The technical difference here is that those layers are distributed, reflecting the network-determined (or estimated) capacity for a timely response.

Aggregation is a complementary perspective on the same issue. Our experience and that of others [Ranathunga et al., 2012] is that the ***Jason*** [Bordini et al., 2007] agents cannot handle high percept update frequencies (actual figures are not very useful because they are inevitably application and platform specific useful), which is typically

manifested by unstable and hard to re-produce behaviour. One approach might be to re-engineer ***Jason*** for higher performance, which although possibly desirable, does not consider whether all those events actually need to be processed at the BDI, that is the deliberative, level. In both theory and practice, cognitive architectures use layers to aggregate *small* observations into *bigger* ones. This can be characterised as inference or situational awareness, depending on perspective, but the overall effect is that minor observations are somehow collected, correlated and classified into less minor observations, subject to some degree of probability that reflects the accuracy of the process. In doing so, the volume of data, which possibly at some level may be labelled "information", is reduced so that frequency of communication is also reduced and the receiving reasoning process is presented with synthesized knowledge reflecting some kind of summary of the situation, rather than having to carry out that process itself. It is a fundamental design challenge, perhaps reflecting the principle of so-called sliding autonomy, to decide which levels should make which decisions, whether those strata are fixed and if not, how those divisions may be determined, or negotiated, in live situations.

We believe that performance is a many-faceted issue in this context and XMPP server throughput, and to a lesser extend network latency, whilst significant, are not the only factors, and it is as much the other components, but especially the deliberative architectures that we choose to use, the rate at which they can absorb percepts and the rate at which they can make effective decisions. This in turn is significantly affected by the level at which it is demanded they reason. Thus, the second mitigation is the relative ease with which new event processors can be added to this architecture, by subscribing them to existing feeds and publishing their results to existing consumers, through which it becomes possible to balance the factors of event rate, information level and network latency to achieve performance targets.

Distribution

Observations regarding distribution are relatively brief because, like accessibility, it could be viewed as having been demonstrated in principle, but like scalability, more is needed for it to be demonstrated with confidence. Since the XMPP message transport layer is directly built on HTTP, and since XMPP has been used for some years to support Internet Messaging in various guises (Microsoft Messenger, Google Talk, etc.),

the mechanism has been demonstrated both to distribute and to scale. We have used the BSF in the context of a distributed sensing project, but the case studies reported here have only been run in the same local area network.

3.4 Related Work

AI research has paid substantial attention to how agent behaviour should reflect a response to something sensed from the environment in which it is situated. Cognitive architectures analyse this information, make decisions, and carry out planning to determine the next behaviour to execute. In a dynamic environments, SOAR [Group, b] is a well known example of a classical symbolic reasoning architecture. However, it is also known as rather heavy-weight and can hardly be expected to respond in real time. There is also a range of well-known reactive architectures, including subsumption [Brooks, 2001], Finite State Machines (FSMs) [Brooks, 1991, Bryson, 2003], Basic Reactive Plans (BRP) [Fikes et al., 1972, Bryson, 2003], and POSH plans [Brom and Bryson, 2006], amongst others. Any of the above, perhaps bar SOAR, are suitable decision-makers for avatars in virtual environments, but our choice from among goal-driven approaches, is the popular the Belief-Desire-Intention (BDI) architecture [Rao and Georgeff, 1995]. Beliefs here refer to knowledge about the world in agent's mind, desires are objectives to be achieved, and intentions identify the actions chosen by the agent as part of some plan to help it achieve a particular desire [Mascardi et al., 2005].

A distinct line of research has highlighted the notion of the environment programming in multiagent systems [Ricci et al., 2011]. According to [Ricci et al., 2011], agent programming should have a balance between the agent itself and its environments in order to achieve a high level of intelligence. This perspective reflects the idea that the environment becomes a meaningful place to support the agent's abilities with many functionalities, rather than the traditional view in which it is simply a place that the agent senses and acts upon.

In this context, there is a fair body of research into the deployment of an embodied artificial intelligence using the above cognitive architectures in rich environments. For example, Bogdanovych et al [Bogdanovych et al., 2008a] introduce the 3D Electronic Institution, or Virtual Institution (VI), which is a virtual world with normative regulations governing interactions between participants and environment. They also also

propose the introduction of virtual characters capable of learning [Bogdanovych et al., 2008b] and the use of VI as environments for imitation learning, providing for the enhancement of virtual agent behaviour by learning from human users or other software agents. Later work from this group puts forward a teaching mechanism, so that the virtual character may become more believable [Bogdanovych et al., 2008c]. Although this work is amongst the most developed in the use of Second Life, it offers rather less on the matter of agent-environment programming and the role of cognitive architectures, because of its focus on the regimented normative environment and virtual characters that learn.

Veksler [Veksler, 2009] demonstrates integration between ACT-R [Anderson et al., 1997, Group, a] and Second Life via a HTTP web server. In this work, all information is gathered by a 3D object, which is attached to the avatar. It scans the environment around the avatar within a certain radius, and sends what the scanner senses to a dedicated web server. The ACT-R module is separate from the Second Life environment, but capable of communicating to the web server. By means of HTTP request to the web server, the decision-making module collects sensing data and executes a 'perceive–think–act' loop. In the end, the decision including motor actions goes back to the intermediate web server, and are then applied to the avatar. Another notable work is [Ranathunga et al., 2011], in which a ***Jason*** agent supplies the reasoning for a virtual agent in an environment provided by Second Life, supported by an external data processing module that handles environment sensing. In the same manner as above, through an attached 3D object, which serves as a virtual sensory system, sets of perceptions generated by the data processing module are delivered to ***Jason*** agent, which then deliberates. The results of the reasoning are communicated back to the Second Life avatar, and the action is realised, changing the state of the environment. The scenario in this case is the playing of a football game. This work demonstrates the utilization of an event recognition platform [Ranathunga et al., 2012] not only to enhance the perception capability, which becomes a source of better reasoning, but also to retrieve more accurate domain-specific information from low-level data.

In comparison with the above systems, which are quite tightly coupled, other approaches also exist, that aim for a more general integration between cognitive agents and virtual environments. There are (at least) three representative systems, with similar objectives to ours, against which we contrast what has been presented here: GameBots [Adobbati et al., 2001], Pogamut [Gemrot et al., 2009], and CIGA [van Oijen

et al., 2012].

Gamebots [Adobbati et al., 2001] has much in common with the virtual agent component in our system, being a kind of programmable agent controller, integrated with the 3D video game Unreal Tournament (UT), in order to create autonomous bots that interact with human players as well as other bot players. The bots are able to sense and act directly from the environment, via TCP/IP socket communication. Gamebot 'agents' appear to be limited to reactive behaviours, whilst the system as a whole only functions with one game engine, namely Unreal Tournament.

Pogamut [Gemrot et al., 2009] incorporates an interface layer between Unreal Tournament and the decision-making agent, by means of TCP/IP sockets. The role of this component is rather like that of the ***N-Jason*** agent in our system, in that it has the task of perceiving the environment, interaction with the environment, and decision making, for which it uses the POSH reactive planner [Brom and Bryson, 2006]. As with Gamebots, Pogamut has seen substantial up-take, from student projects to complex research projects, thanks to their approach that allows greater flexibility in the development of the high level of autonomy in virtual agents. Nevertheless, it still has a high dependency on the particular environment of UT, and on a particular programming language, namely Java.

CIGA [van Oijen et al., 2012] also has numerous similarities with our framework in that it aims to resolve the coupling problem between agent and virtual environment. This it does by means of two interface layers and an ontology model: (i) physical interface layer to connect to a environment (game engine), and (ii) cognitive interface layer to connect to a multiagent system, corresponding to the Virtual Agent and the ***Jason*** Agent, respectively. The use of ontology model, containing pre-defined ontologies to make a contract between agent and game engine even though they are situated in a specific domain, eases the interpretation of perception and behaviour execution. This architecture offers fair accessibility, and could in principle support distributed execution, thanks to the use of socket-based communications, but this would require careful manual configuration. In this respect, CIGA is the closest to our proposal, but the dependence on a relatively low-level and inflexible network layer seems likely to inhibit distribution and scalability.

To summarise, the short-comings we observe in the above lie in their tight integration of the components, leading to an effectively closed, single platform sys-

tem. Thus, they do not have the flexibility necessary for distributed software systems. They are tightly coupled by the communication protocol as well as ontology, so that adding/removing a software component is challenging, and deploying such platforms more widely is in general difficult. This is only exacerbated – or probably even rendered impossible – if it is desired to incorporate components in different programming languages or that only run on another operating system.

Earlier, we noted the lack of any comprehensive performance evaluation of XMPP, as far as we can find, in the academic literature. Linden Labs have apparently carried out an evaluation of various message passing protocols[6], noting that the Advanced Message Queueing Protocol [Committee, 2012] implementations demonstrate good single-host performance, but lack figures on maximum capacity when clustered, or the value of clustering. XMPP appears in the list, but was not evaluated for lack of time. Several other protocols are eliminated for not meeting their requirements, but unfortunately there is no definitive conclusion.

3.5 Summary

In this chapter, the Bath Sensor Framework has been introduced as a middleware for decoupling cognitive agents and virtual environments. Also, a case study is presented using the framework for linking heterogeneous software, ***N-Jason*** BDI type cognitive architecture and *SecondLife* which is representative rich 3D virtual environment. From the study, it seems clear that BSF has some useful advantages as an integration middleware. Firstly, it offers good accessibility, because of the simplicity in protocol, and ease of both connection and use. Secondly, in respect of speed and reliability, it inherits from XMPP, so that it is able not only to communicate in (soft) real time but also transfer whatever data in the form of open XML, which may become the basis of interoperability in cross domain applications. Finally, it enables distribution of components, so that it contributes to effective data transport mechanism such as 1-to-1, 1-to-many, or many-to-many, from anywhere to anywhere. As a result, through the use of the BSF, the system as a whole has the potential for flexibility and extensibility.

In the following chapter, we have extended the framework to operate in conjunction

[6] `http://wiki.secondlife.com/wiki/Message_Queue_Evaluation_Notes`, retrieved 20120416, last updated 2010.

with an institutional framework, so that the behaviour of an agent in a virtual environments can be governed by norms.The notion of institution, as a set of rules for a particular agent society, is appropriate for the regulation of behaviour of virtual agents in a particular situations, by saving the need to incorporate behaviour for all circumstances in the agent themselves.

Chapter 4

Governing Virtual Characters Behaviour with Norms

In the previous chapter (Chapter 3), we present how (BDI type) cognitive reasoning agents can control virtual characters behaviour through the services of a middleware. In this chapter, we present **D**istributed **N**orm-**A**ware **A**gent **A**rchitecture (**DNA**^{3}) as a model of a socio-cognitive reasoning framework for virtual characters by the use of insitutions [Cliffe et al., 2007] on top of the above integration, thus enabling cognitive reasoning for virtual characters.

4.1 Introduction

Recalling the main objective, the thesis centres on reasoning for virtual character behaviour with (social) norms from a *socio-cognitive* perspective [Stein, 1993, Bandura, 2001, Akgün et al., 2003] which emphasises the social influences on individual cognitive processes in human choices. This characteristic of the theory in turn gives rise to the necessity of incorporating social as well as cognitive processes in the selection of behaviour when individuals are engaged in social settings.

One approach to the realisation is to utilise institutional models [Cliffe et al., 2007] as a social reasoning process in conjunction with cognitive reasoning virtual charaters. The institutional models – also called normative frameworks – are seen as an effective way to capture the salient elements of human social structures [Boella et al., 2006] and

can equally be applied in (participatory) VEs to provide IVAs with adequate knowledge of human social mores, or indeed the rules of whichever context in which we wish them to be able to behave properly. Thus, the use of institutions is a way to achieve the appropriate recognition of complex situations and provide guidance on the consequent choice of action(s) depending on the context the virtual characters encounter.

In response, this chapter introduces a computational model, **D**istributed **N**orm-**A**ware **A**gent **A**rchitecture (**DNA**^{3}), in order to facilitate the socio-cognitive approach in reasoning virtual character behaviour with (social) norms. We attempt to establish **DNA**^{3} by the additional integration of institutions [Cliffe et al., 2007] on top of coupling BDI-type cognitive agents and virtual characters shown in Chapter 3. Given (**DNA**^{3}), we intend to answer the following questions as noted in Chapter 1 through **DNA**^{3}and its examples: (i) how institutions carry out reasoning about situationally appropriate norms under the combination of BDI-type agents and virtual characters, (ii) how these norms can be adopted in a virtual agent's mind and (iii) how these norms can be associated with agent decision making process.

We start the chapter by explaining socio-cognitive theory [Stein, 1993, Bandura, 2001, Akgün et al., 2003] through the survey on psychology and social science literatures in Section 4.2 as a theoretical foundation. Then, in Section 4.3, we present the introduction to computational model of institutions Cliffe *et al.* introduce [Cliffe et al., 2007], where social cognition in virtual environments takes place. Afterwards, we propose **DNA**^{3} as a socio-cognitive reasoning framework for virtual character behaviour in Section 4.4. This includes an informal explanation of the socio-cognitive reasoning process in **DNA**^{3} (Section 4.4). Two illustrative examples are given in Section 4.5.

4.2 Socio-Cognitive Perspective on Human Choices

This section identifies a theoretical background, the socio-cognitive perspective on human choices, as it appears in psychology and social sciences. To begin with, a background that what motivates an advent of the socio-cognitive perspective in human choices is described in Section 4.2.1. Afterwards, a brief overview about socio-cognitive decision making process is introduced in Section 4.2.2. We particularly emphasise the role of institutions as a means to mimic the social reasoning process in the human society.

Note that the material presented in this Chapter is mostly grounded upon the literature of Anderson [Anderson, 2005], Bandura [Bandura, 2001, Bandura, 2005], Akgün, [Akgün et al., 2003] and Stein [Stein, 1993, Stein, 1997].

4.2.1 Cognitive Psychology: Background

A decision making process organising human thought and action has been given much attention in psychology. Traditionally, it has been seen as a simple input-out model inspired by behaviouristic principles. Accordingly, human choices are shaped and controlled by stimuli from external environments in both a mechanical and automatic way in this linear model [Bandura, 2001]. This perspective has been rapidly supplanted by more dynamic and complicated models in modern psychology. A reasoning process can be regarded as a result of simultaneous and interactive computations carried out by several subpersonal components which orchestate a series of actions [Harré, 1983, Bandura, 2001].

Cognitive psychology contributes to understanding the underlying mechanism of human decision making in accordance with the development of modern psychology [Neisser, 1976, Anderson, 2005]. It successfully illustrates such simultaneous and active involvements of diverse operations in human mind with a concept of 'human cognition'. According to Hilgard [Hilgard, 1980], the term 'human cognition' means '*the process of knowing*' about newly incoming information people are receiving from the external environment. This process involves several simultaneous and/or consecutive brain activities such as processing, codifying, representing, interpreting, storing and retrieving the information subject to the occurrences in external environments [Stein, 1993, Anderson, 2005] in order to '*make sense*' of the observations. More details are decribed in the succeeding section.

Cognitive Structure

The cognitive structure is a form of complex pattern in the human brain, which is constructed by recognising, arranging, systematising and storing stimuli perceived by individuals. This cognitive structure mainly plays an important role in making sense of newly incoming information when people are receiving a series of stimuli from external environments. If this new information is too much to process within an individual's

cognitive capacity, then the cognitive structure contributes to economise their cognitive capacity [Anderson, 2005].

The cognitive structures starts the task (i.e. to process newly income information) just after the stimuli are received and stored in the sensory memory via the heuristic search mechanism governed by *"gestalt principles"* [Anderson, 2005]. This registered information subsequently enters into the short-term memory, then those incoming information is identified by a pattern recognition techniques e.g. *"recognition", "heuristic search", "pattern recognition"* and *"serial recognition"* [Simon, 1990]. These techniques are essentially carried out based on: (i) the search for combinations of features, (ii) the situational context and (iii) the attention given [Stein, 1993].

If the information is not identified successfully by the use of cognitive structures, and which is in turn proved as unfamiliar from the existing set of cognitive structure, then it gives rise to more attention leading to the request for more cognitive capacity to process. Depending on the amount of attention given, the cognitive structure processes those unfamiliar information either *"automatically"* (or *"habitually"*) in case that low attention given or *"actively"* otherwise [Anderson, 2005].

Interpretations of stimuli are the final step of the process, which in principle aims to give a meaning in the end. At this stage, those interpretations are stored in the long-term memory by the principle that the schema theories of memory proposed. Actually, the stimuli (or experiences) are interpreted to a large extent and guided by the existing structures in the long-term memory. In other words, schema (also called a *frame*) selects and modifies experiences so that a coherent and consistent representation subject to schema (reflecting existing memory) is formulated.

Thus, the outcome stored in the memory is always under some degree of influence from the existing knowledge individuals possess [Bourne et al., 1987]. These stored interpretations are in turn distinguished between: (i) *"episodic"* memory identifying the spatial and temporal aspect of an experience, (ii) *"semantic"* memory referring to an individual's knowledge focusing on *what it is* and thus (iii) *"procedural"* memory storing the knowledge relevant to *what one can do* [Bourne et al., 1987]. The information those memories hold can be utilised if required during another cycle of processes later on.

Criticism of Cognitive Psychology

Despite the subtantial impact on understanding human behaviour, some criticisms have been raised about cognitive psychology. On the one hand, it is an issue that the rationality of human cognition is bounded. *'Horizon'*, a TV program on the BBC in the UK, introduced an interesting perspective on individual cognitive reasoning process. The program argues that whether or not human choices are clearly rational during decision making (via individual cognitive process) as ordinary people believe that their decision making is rational. Kahneman [Kahneman and Tversky, 1979] confirms that the assumption, human choices are rational, is not true. In effect, almost all human decisions are likely to rely on *intuition* which is '*biased*' by not only inherited traits from human evolution but also previous individual experiences, rather than by '*logical process*' in their cognition. Kahneman claims this tendency in human choices, called choice bias, is mainly caused by the limited cognitive capacity [Simon, 1990] of the human brain. Since on the one hand, the logical reasoning process requires a significant work load under bounded resources, but on the other, it is easy to recall or imagine from experiences, a response, the human brain tends to economise its mental activities by doing heuristic search and re-using preconceived knowledge instead of carrying out additional reasoning [Kahneman and Tversky, 1979].

The limited choice domain where heuristic search or recalled experience take place is another factor in bounding the rationality of human cognition. As shown in the previous section (Section 4.2.1), the representation of the world is actively constructed by (cognitive) processes where the sensory input is transformed, reduced, elaborated, recovered and used [Neisser, 1976]. Human attention is limited in capacity, thus people are inevitably likely to be selective in their perception of the world, in order to prevent of information processing overload. Even worse, only a small part what people observe can be internalised in the cognitive structure due to its limited capacity. In the meantime, those selective and partial observations are transformed sometimes through reduction and elaboration in the human brain [Bourne et al., 1987, Anderson, 2005, Reed, 2012]. As a result, people may not be able to possess the entire representation of the world and experiences. Instead, it is obvious that only a limited knowledge of choice domain is in their mind. Simon argues that limits on computing resources (e.g. power, speed, data set) of intelligent systems (e.g. human brain, computers) lead the systems to have limited rationality due to the approximation methods necessary to

handle the tasks [Simon, 1990]. In accordance, the limited choice domain as well as limited cognitive capacity in human brain consequently constrain the rationality of a human decision when the human choice is adopted on the basis of the cognitive structure individuals have.

The limited comprehension of human behaviour in social situations is seen as the other shortcoming of cognitive psychology, as discussed earlier in Chapter 1. Like to behavioural theory, cognitive psychology relies heavily on studies (e.g. empirical as well as experimental studies) constrained by elaborat artificial settings [Neisser, 1976, Anderson, 2005]. In other words, the cognitive approach only pursues the investigation of the human mind and its inner working mechanisms without consideration of external influences. This results in the social influences on the human mind not being given due recognition, which gives rise to less comprehension of human behaviour, especially when an explicit causal relation between 'thought about what to do next' and 'actions that individuals can actually pursue' is not seen.

This non-causality in human choices reminds of what Bandura raises in his literature [Bandura, 2001] that "*People do not live their life only in individual autonomy*". People are sometimes likely to give up their individual autonomy during their decision making process when they are situated in the society where a number of individuals are connected by interactions [Simmel and Wolff, 1950]. Although an explicit causality is not seen between individuals' thoughts (e.g. what people really want to do) and obligations (e.g. what should be done), human behaviour can be affected by social forces when the decision that gives up the autonomy is more socially worthy (or fit to moral purpose) in their interactions.

This nature teaches us that people have an ability to make a judgement of the rightness and wrongness by conducting an evaluation of personal standards and social-situational circumstances [Bandura, 2001]. As described above, cognitive structure is a complex knowledge set ordinarily constructed by the (cognitive) processes of '*personal*' collections and experiences. Suppose that a sensory input is personal experiences in the social context. Then individuals internalise the input information as a set of recipes about '*the way things are* and the '*the way things should be done*' [Stein, 1993, Stein, 1997, Anderson, 2005] – which are called preconceptions – in the cognitive strucuture. When they are situated in a similar social context subsequently, they tend simply to recall those preconceptions in order to interpret either situations or asso-

ciated responses from others (or both) so as to behave adequately [Weick, 1979, Weick, 1988]. However, such preconceptions are a personal prior knowledge which can be seen as a '*private sense*' [Stein, 1993] thus the choices based upon preconceptions may not be accepted as appropriate in relation to the situations. In order to judge the adequacy of the behaviour determined by cognitive reasoning process(es), it is desirable to take into account a standard of behaviour in a '*common sense*' with a personal standard, which is a shared common knowledge by members of a society, such as norms addressing what conduct ought to be in that circumstance [Gibbs, 1981].

In conclusion, cognitive psychology has led a contribution to the understanding of the internal mechanism that how human brain functions during the decision making process. However, it shows a shortcoming with respect to (i) the bounded rationality of human cognition and (ii) the limited comprehension of human choices under social forces. Subsequently, such a recognition of the limitaions enables to take a different route in theorising underlying mechanisms in human mind, towards a socio-cognitive approach in order to comprehend human behaviour better.

4.2.2 Socio-Cognitive Theory

As discussed in the previous section, cognitive psychology has limitations, that makes it insufficient to illustrate the internal mechainsm of human mind, specifically under societal structures and its sub-systems [Stein, 1997, Anderson, 2005]. As a result, sociology in response attempts to comprehend human behaviour in a macro-analytic way, so that the determination of human choices and its associated behaviour are socially interdependent and such that it is an outcome of the interplay between individual knowledge and social structures [Cook and Yanow, 1993, Easterby-Smith, 1997, Bandura, 2001].

A socio-cognitive theory is in principle grounded on the above hypothesis, the pursuit of the incorporation of a social as well as a cognitive dimension in human choices. The cognitive dimension stands for an individual knowledge structure that personal collectives have constructed in their mind as seen in Section 4.2.1. Similarly, social dimension refers to an intersubjectively-shared knowledge structure about "*the way things are and the way things should be*" [Stein, 1997]. Human behaviour is closely related to this knowledge in social situations, which takes responsibility for the interpretations of that deontic forces at play in that situation, as cognitive structure does during the cognitive process. In this sense, the social dimension is seen as a result of

social structure and processes driven by collectives of people, called 'social cognition' that can be compared to human cognition on the individual level.

In general, the notion of 'cognition' refers to the process of 'knowing' about newly perceived information as noted in Section 4.2.1. Given this concept, cognitive psychology defines human cognition as a mechanism activated for the processing and storing information in the human brain [Anderson, 2005]. Likewise, social cognition is seen as processes to construct socially-shared knowledge structure through functions similar to those in human cognition, such as acquisition, storage, transmission and manipulation of social information [Larson and Christensen, 1993].

Social cognition mainly investigates the (conscious/unconscious) human information processing in complex social interactions [Gioia and Sims, 1986] taken by individuals with one another. This is closely related to the identification of the way how people interpret and construct their social reality and all aspects of reasoning processes [Weiner et al., 1983]. With this aim, social cognition develops the socially constructed knowledge structure through social activities amongst people and a collective of people in the society. Such information processing of social stimuli produces a knowledge structure representing a 'common sense' in the form of norms, rules or culture [Klimecki and Lassleben, 1998]. Similar to cognitive process, these knowledge structures operate as a central means to interpret social interactions in which individuals are currently engaged [Wyer Jr and Srull, 1984].

Recalling the standpoint of the socio-cognitive approach, it is not unusual that a causal relation might not be seen between 'thoughts about doing something' and 'actions actually allowed' in the human mind, due to the social context in which people are embedded [Stein, 1997]. Individual perceptions are interpreted on the basis of the cognitive dimension so that thoughts about what to do next in relation to those perceptions are independent from the external influences. In other words, the interpretation relies entirely on an individual knowledge structure in the human brain representing a 'private sense'. But actions actually taken may not be in the category of those thoughts (i.e. desires) people initially have in their mind, when the situation they now encounter sometimes requires a special consideration. Due to the external deontic forces the special consideration imposes, the actions people can actually pursue can be restricted and/or organised by the consideration, rather than desires (i.e. thoughts driven by individual cognitive processes). In consequence, there might not be an explicit causality

between individual thoughts and embodied actions, which cannot be explained or understood by cognitive psychology at all. In this case, social cognition could be a clue for the identification of the reciprocity between individual desires and situationally appropriate actions being taken.

In review, we believe it is important to consider both cognitive and social dimension during the decision making. This approach provides a complementarity for the interpretation of the reciprocity between desires and actual choices in the social context, by the formation of social phenomena through interaction processes [Stein, 1997]. Within this sense, the fertilisation of social cognition is essential for socio-cognitive decision making. We now introduce the institution where social cognition takes place in conjuction with individual cognition in the following sections.

Institutions for Social Cognition

Social cognition is achieved by a collection of people in a social context through the processes of interaction amongst them. This results in the knowledge structure representing the norms, rules or cultures which affect behaviour during interactions by acting as a frame to interpret a meaning of reality from those interactions [Duncan, 1979, Kim, 1998, Nonaka and Takeuchi, 1995, Akgün et al., 2003]. An institution is a socially constructed belief system functioning as described above [Stein, 1993, Stein, 1997]. As in the way of human cognition (e.g. cognitive process and structure), the institution takes responsibility for both process and structuring of social information, following the operations that how the social information is stored, organised, retrieved and selected [Douglas, 1986].

The institution is a structural discipline in the processing of information [Stein, 1993]. It governs the creation of meanings so as to pattern human actions/interactions at various social levels – which can be seen as a social process. Note that the actions/interactions are not actually taken by the instition themselves during the process of interactions. Instead, the institution acts through the mediator (which is the individual and their cognitive process) by the exertion of influence on them. The institution is then influenced by the knowledge those individuals have induced [Douglas, 1986, Akgün et al., 2003]. Subsequently, the institution governs the gathering and processing of the information [Stein, 1997].

As a result, this social process produces the social structures whose contents are

normative patterns defining expected modes of action or expected modes of social relationship [Parsons, 1940]. In other words, the institution has a set of norms describing socially-shared expected behaviour in particular situations (or circumstances) agreed by all members of society [Gibbs, 1981, Schotter, 2008]. Given such a set of norms, the institution interprets and organises human interactions by means of normative patterns representing "*the way things are*" and "*the way things should be*" [Stein, 1997], which can be seens as a social structure.

In principle, norms in the institution work as expectations in social relations. According to Wallace [Wallace, 1983], the norms brings the 'common sense' to the interpretation and choice of behaviour for participants in four ways that: (i) from the roles the individuals address ("*actor expectations*"), (ii) from the context, in particular the time ('when') and the space ('where') ("*situation expectations*"), (iii) from the experiences of causal effects ("*response expectations*") and (iv) from the values which motivate, assess and validate actions taken ("*consequence expectations*") [Wallace, 1983]. This in turn promises the improved rationality in actions/interactions with regards to (i) the adequacy in behaviour subject to the roles, (ii) the accuracy in time and space, (iii) the mutuality in response and (iv) the trustworthiness of the actions undertaken [Weber, 2009].

In conclusion, the institution is a socially constructed belief system where social cognition takes place. The institution is on the one hand in charge of the construction of social structure, consisting of normative contents through the processing a collective of people in the interactions. On the other hand, it interprets the social reality with normative patterns, so as to organise human behaviour in the social context. With the nature of norms addressing appropriate actions subject to the situation, the institution leads the enhancement of rationality in social behaviour of individuals by exerting norm influences (which imply 'common sense') on individual cognitive reasoning.

4.2.3 Discussion

Although cognitive psychology has contributed to the understanding of human behaviour, it still has limitations in comprehending human behaviour in a social context. As a resolution of this tension, this section presents a socio-cognitive theory, which is in the pursuit of the incorporation of a cognitive as well as a social dimension in human choices, in order for the better comprehension of human behaviour in the social

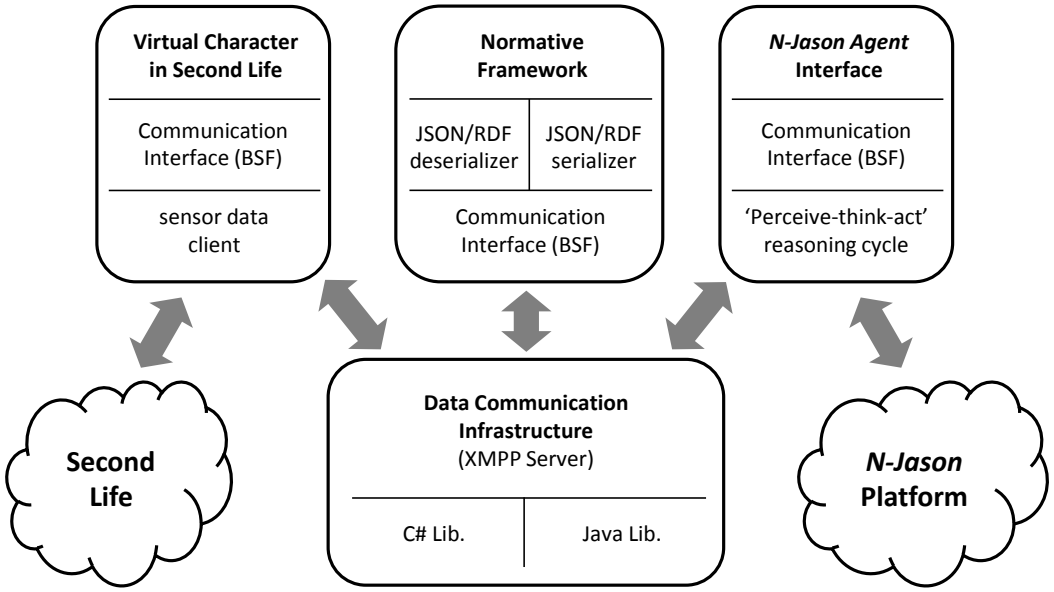

Figure 4-1: ***DNA***^{3} *System Overview*

context. The institution as a means for social cognition plays an important role in this theory by the exertion of normative influences on individual cognitive reasoning.

In response, we propose the Distributed Norm-Aware Agent Architecutre (**DNA**^{3}) shown in Figure 4-1 in order to design and simulate virtual characters' behaviour with norms in the rest of this chapter. Recognising the importance of norms in human social intelligence, **DNA**^{3} includes institutional models on top of the Intelligent Virtual Agent (IVA) which couples cognitive reasoning agents and virtual characters described in Chatper 3. In doing so, virtual characters can show a better quality of norm-aware behaviour by means of socio-cognitive reasoning which considers social structure as well as individual cognition, in particular in the situation that social activities are necessary. In the next section (Section 4.3), we introduce the formal and computational model of institutions as a preliminary to **DNA**^{3}. Afterwards, the core of **DNA**^{3} is presented in Section 4.4.

4.3 An Institutional Model in MAS

Institutional models – also known as normative frameworks – are a kind of external source of knowledge for delivering norms to intelligent (virtual) agents in MAS research. It is a set of rules for being able to govern the agent society. These rules can

be seen as situation-specific norms resulting from reasoning about the current social context, rather than just a hard-coded repertoire of reactions such as those in static expert systems. It describes not only correct and incorrect actions but also norms such as obligations, while maintaining a record through its internal state, that evolves according to events captured from the external world. Inside the institutional models, the information about 'when' and 'what' is the correct (or incorrect) action to be achieved by agents, evolves over time according to the given social context in the models. Once these norms are issued, agents are able to carry out 'rational' decision making with norms which can lead to a 'right' and 'situation-specific' behaviour created by situational awareness in social settings.

We adopt Cliffe's institutional framework [Cliffe et al., 2007], which provides the function of regulatory governance of agents through its capacity for social reasoning. The regulation takes the form of permissions and obligations that an agent is free to follow or ignore, but the latter may have social consequences, such as ostracism [Perreau de Pinninck et al., 2007], for example – although we do not explore this issue further here. The framework provides a formal action language Inst*AL* to specify norms describing social interactions between agents and (or) environments in the context of the institutions. Then the formal framework is translated to a computational framework based on Answer Set Programming (ASP) [Gelfond and Lifschitz, 1991], which enables the reasoning about the current social context described in the institution.

The actual modelling of the institutions is achieved by Inst*AL* , an institutional action language proposed by Cliffe *et al.* [Cliffe, 2007]. Underpinning the action language is a formal mathematical model and an equivalent computational model implemented in $AnsProlog$. We now briefly give overview of both models adapted from the citations given in the following section.

4.3.1 Formal Model

Inst*AL* 's underlying principle is the interpretation of exogenous events in the context of the institution, using [Searle, 1995]'s principle of conventional generation. The normative effects of these events are recorded in the institutional state.

Two types of events are defined in the model:

- *external events* ($\mathcal{E}_{ex}$) capture the events happening in the VE, and

- *institutional events* ($\mathcal{E}_{inst}$) are the interpretation of external events in the institutional context.

Following the "count as" principle introduced in [Searle, 1995], external events can be interpreted into the corresponding institutional events. For example, an external event "say hello" occurring in VE, counts as an institutional event "request chatting". Institutional events are divided into two groups: institutional actions ($\mathcal{E}_{act}$), indicating changes in the institutional states, and violations ($\mathcal{E}_{viol}$), generated by performing non-permitted actions or non-satisfaction of obligations.

The institutional state is represented by a set of facts, called fluents $\mathcal{F}$. At any given instant, their presence denotes the veracity of the fact and the absence otherwise. Therefore, a state formula is a combination of positive or negative fluents: $\mathcal{X} = 2^{\mathcal{F} \cup \neg \mathcal{F}}$. Different aspects of the normative state are denoted by subsets of $\mathcal{F}$ which may be separated into *domain fluents* and *normative fluents* further as below:

- *domain fluents* ($\mathcal{D}$) describe domain-specific properties,
- *power* ($\mathcal{W}$) indicates some event is empowered and may so bring about institutional change,
- *permission* ($\mathcal{P}$) denotes an event can be performed without generating a violation, and
- *obligations* ($\mathcal{O}$) specify that an event must happen before the occurrence of deadline (e.g. a timeout), otherwise a violation is generated.

In practice, the normative fluents (e.g. $\mathcal{P}$, $\mathcal{O}$) represent the normative consequences of particular behaviours which should be achieved by virtual agents at a certain social context. For example, the form of the normative information is represented as: $\texttt{obl}(\texttt{act}, \texttt{deadline}, \texttt{violation})$ or $\texttt{perm}(\texttt{act})$, which means an agent *X* is obliged to carry out action *act*, or an agent *X* is permitted to perform action *act*, respectively.

By observing a trace of exogenous events, the VE's institutional states evolve accordingly. To this end, two transformer functions are provided:

- the generation relation ($\mathcal{G}$), generates institutional events from the occurrence of external/institutional events subject to conditions on the state, and

$\mathcal{I} = \langle \mathcal{E}, \mathcal{F}, \mathcal{G}, \mathcal{C}, \Delta \rangle$, where

1. $\mathcal{F} = \mathcal{W} \cup \mathcal{P} \cup \mathcal{O} \cup \mathcal{D}$
2. $\mathcal{E} = \mathcal{E}_{ex} \cup \mathcal{E}_{inst}$ with $\mathcal{E}_{inst} = \mathcal{E}_{act} \cup \mathcal{E}_{viol}$
3. $\mathcal{G} : \mathcal{X} \times \mathcal{E} \rightarrow 2^{\mathcal{E}_{inst}}$
4. $\mathcal{C} : \mathcal{X} \times \mathcal{E} \rightarrow 2^{\mathcal{F}} \times 2^{\mathcal{F}}$ where $C(\phi, e) = (\mathcal{C}^{\uparrow}(\phi, e), \mathcal{C}^{\downarrow}(\phi, e))$ where
   (i) $\mathcal{C}^{\uparrow}(\phi, e)$ initiates fluents
   (ii) $\mathcal{C}^{\downarrow}(\phi, e)$ terminates fluents
   (iii) with ϕ a condition on the state $\phi \subseteq 2^{\mathcal{F}} \cup 2^{\neg \mathcal{F}}$ and $e \in \mathcal{E}$
5. $\Delta \subseteq \mathcal{F}$
6. State Formula: $\mathcal{X} = 2^{\mathcal{F} \cup \neg \mathcal{F}}$

Figure 4-2: *Formal specification of the institution*

- the consequence relation ($\mathcal{C}$), updates institutional states by adding or removing fluents, subject to the occurrence of some event and other conditions.

Therefore, given an event trace and initial state Δ of an institution, the corresponding institutional model, i.e a sequences of corresponding states, can be generated.To summarize, an institution is a tuple $\mathcal{I} := \langle \mathcal{E}, \mathcal{F}, \mathcal{G}, \mathcal{C}, \Delta \rangle$ and the main features are defined in Figure 4-2.

All these elements are specified in the institution by Inst*AL* . The actual operation of the reasoning about normative consequences, subject to an observed but incomplete trace of $\mathcal{E}_{ex}$, is accomplished by answer set solver, just after the translation of formal institution model to a corresponding computational model using ASP. In the reasoning process, answer set solver performs traversing all cases with descriptions and constraints derived from $\mathcal{G}$ and $\mathcal{C}$, and find the most adequate answer in all answer sets afterwards. More details are described in the following section.

4.3.2 Computational Model

The formal model of institutions described above can be translated to a corresponding computational model using answer set programming.

ASP [Gelfond and Lifschitz, 1991] is a declarative programming paradigm under

the answer set semantics. Instead of designing solution to a problem, ASP only requires the description and constraints to the solution, and then finds the solutions. The basic elements are atoms, that can be given a value, true or false. Atoms can also be negated by *negation as failure*. The general form of an ASP rule is syntactically like that of a Prolog term, comprising a head and and a body, such that the truth of the former is implied by that of the constituents of the latter. Rules with no head express *constraints*, indicating undesirable rules that solutions should not satisfy. Thus, an ASP program is a conjunction of rules. Solutions are found by assigning values to atoms in order to satisfy all the rules stated in the program in a minimal and consistent fashion. Each such solution is an answer set.

The computational model of an institution comprises:

- base component, initiates/terminates fluents, generates violation events if necessary,
- time component, defines time predicates as observed event and forms time sequence, and
- institution-specific component.

The main ASP atoms in this computational model are:

1. `fluent(p)` for a fluent $p \in \mathcal{F}$ of an institution,
2. `event(e)` for an event $e \in \mathcal{E}$,
3. `evtype(e,obs)` to denote an observed (exogenous) event $e \in \mathcal{E}_{ex}$,
4. `evtype(e,act)` to denote an institutional action $e \in \mathcal{E}_{act}$,
5. `evtype(e,viol)` to denote a violation event $e \in \mathcal{E}_{viol}$,
6. `initiated(p,T)` and `terminated(p,T)` denote a fluent p is initiated/terminated at time T,
7. `occurred(e,T)` to denote the occurred event e in the institution at time T, and
8. `holdsat(p,i00)` denotes that the fluent p is true in the institutional model at time instant `i00`.

$$p \in \mathcal{F} \Leftrightarrow \texttt{fluent(p)}. \tag{4.1}$$
$$e \in \mathcal{E} \Leftrightarrow \texttt{event(e)}. \tag{4.2}$$
$$e \in \mathcal{E}_{ex} \Leftrightarrow \texttt{evtype(e, obs)}. \tag{4.3}$$
$$e \in \mathcal{E}_{act} \Leftrightarrow \texttt{evtype(e, act)}. \tag{4.4}$$
$$e \in \mathcal{E}_{viol} \Leftrightarrow \texttt{evtype(e, viol)}. \tag{4.5}$$
$$\mathcal{C}^{\uparrow}(\phi, e) = P \Leftrightarrow \forall p \in P \cdot \texttt{initiated(p, T)} \leftarrow \texttt{occurred(e, T)}, EX(\phi, T). \tag{4.6}$$
$$\mathcal{C}^{\downarrow}(\phi, e) = P \Leftrightarrow \forall p \in P \cdot \texttt{terminated(p, T)} \leftarrow \texttt{occurred(e, T)}, EX(\phi, T). \tag{4.7}$$
$$\mathcal{G}(\phi, e) = E \Leftrightarrow g \in E, \texttt{occurred(g, T)} \leftarrow \texttt{occurred(e, T)}, \texttt{holdsat(pow(e), T)}, EX(\phi, T). \tag{4.8}$$
$$p \in \Delta \Leftrightarrow \texttt{holdsat(p, i00)}. \tag{4.9}$$

Figure 4-3: *Translation of institutional rules into* $AnsProlog$

A formal model of an institution can be translated to an ASP program by the mapping from the formal model to ASP atoms according to Figure 4-3. For all exogenous, institutional and violation events, lines 4.1 to 4.5 maps them into ASP atoms. Lines 4.6 to 4.9 show the framework-specific translation rules. As introduced in the previous section, the set of state formula $\mathcal{X}$ denotes all possible states characterised by the combination of positive ($\mathcal{F}$) or negative fluents ($\neg\mathcal{F}$). For a given condition $\phi \in \mathcal{X}$, the corresponding institution event $g \in E$ is generated ($\texttt{occurred(g, T)}$) at time T subject to some conditions $EX(\phi, T)$ when an event e occurs at time T ($\texttt{occurred(e, T)}$) according to the generation rule $\mathcal{G}(\phi, e)$ (line 4.8). When an institutional event occurs, some fluent $p \in P$ might be initiated ($\texttt{initiated(p, T)}$) (line 4.6) or terminated ($\texttt{terminated(p, T)}$) (line 4.7) at the same time T. The literal ($\texttt{not}$) $\texttt{holdsat(f, T)}$ represents the fluent f holding positive (or negative) at time T. The initial states f of institution at time $i00$ is encoded as $\texttt{holdsat(f, i00)}$.

4.4 Agent Deliberation with Institutions

As introduced in Section 4.2, socio-cognitive theory pursues the incorporation of social dimension and cognitive dimension in order to improve the rationality in human decision making in the social context. Inspired by the theory, this thesis take into account the hybrid approach which combines social reasoning and individual cognitive reasoning in the design and simulation of norm-aware virtual characters behaviour. In the preceding section, we introduced the institutional model which take responsibility of social reasoning as a computational model of social cognition in the socio-cognitive theory. Now we would like to show the way how the social reasoning carried out by the institutional models can be incorporated with individual virtual agents so as to guide virtual characters behaviour with norms.

In this section, we introduce **DNA**[3] which consists of such institutional models supporting social reasoning, and ***N-Jason***[1], a norm aware (BDI-type) cognitive agents supporting individual reasoning. To begin with, the architecture of this framework is presented from an engineering perspective, explaining in detail how the coupling is accomplished between the institutional model and the Virtual Environment (VE) wherein the Virtual Agents (VAs) (controlled by BDI agents) are situated. Subsequently, we discuss from the operational perspective how the mental model of the framework is implemented. To do so, we illustrate the internal mental state changes associated with how an institution and norms affect the individual reasoning process.

4.4.1 Architecture

Existing examples of agent platforms working with VEs [Arcos et al., 2005, Savarimuthu et al., 2008, Bogdanovych et al., 2008b] have quite a high degree of integration, tight coupling, or bespoke software involved. Close integration is potentially beneficial for performance, but potentially problematic when one of the software components changes. Thus, we concluded that it was desirable to decouple agent platform and VE to minimise the impact of change in one on the other, but close enough to deliver acceptable performance and genuinely distributed execution. Furthermore, existing architectural approaches would only get harder to maintain if we sought to integrate a third component – the institution – into the system.

[1] See chapter 5 for more details

We use a lightweight distributed framework, BSF, introduced in the previous chapter (Chapter 3), originally conceived for distributed sensors, as a means to provide communication and decoupling. It presents a simple and flexible programming environment, using publish and subscribe streams, supported by XMPP server technology, to connect multiple software components that can operate on one computer or over a local or a wide area network. As mentioned in Chapter 3, we realise the system of cognitive reasoning virtual characters by setting up a ***N-Jason*** platform for the cognitive agents and the libOMV [OpenMetaverse Organization, 2012] interface to the Second Life server as clients to an XMPP server. Subsequently, we can incorporate the institution component by connecting it using the same pub/sub mechanism.

The system overview of **DNA**^{3} is shown in Figure 4-4. The diagram is more general than the system we describe here, in that it refers to a connection manager (libOMV, in this case) and virtual environments (VEs)(Second Life, in this case). Different VEs can be connected, as can device interfaces with the real world, such as Kinect, for gesture recognition, both of which have been done. Thus, it is possible to support all four combinations of human/IVA interaction within a VE.

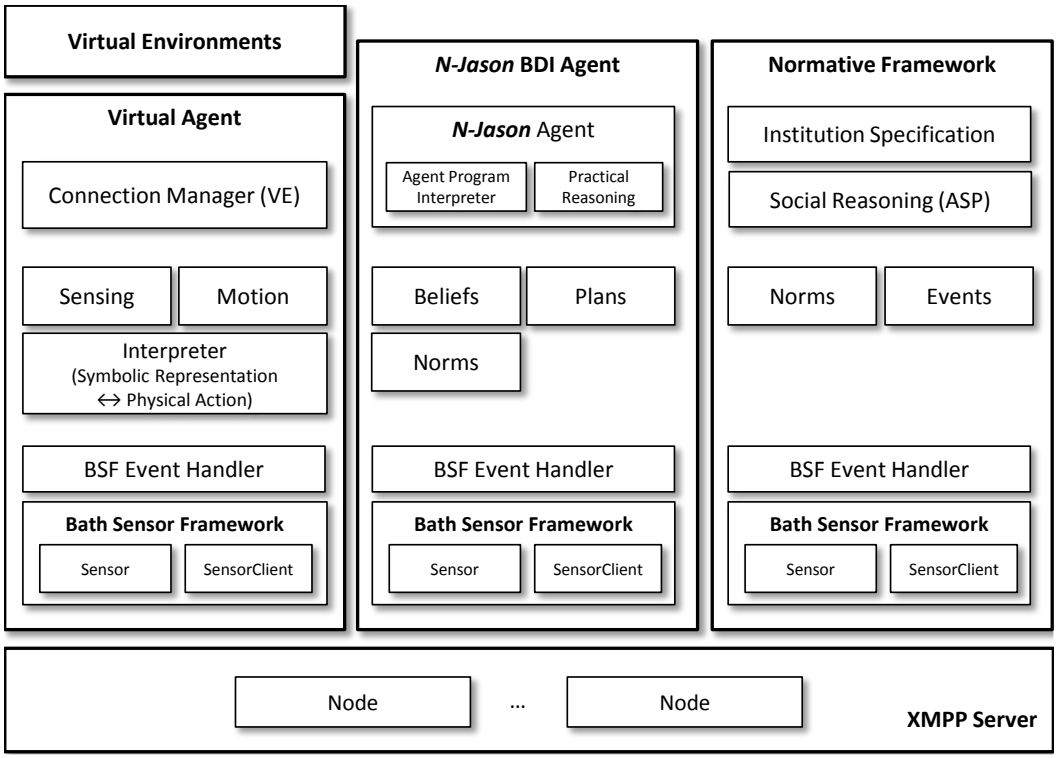

Figure 4-4: *High Level Architecture of* ***DNA***^{3}

As shown in Figure 4-4, the **DNA**^{3} system is composed of three software components as follows:

1. the normative framework, which formulates and delivers norms corresponding to environmental events delivered by agents,

2. the virtual character, that resides in the VE and being capable of sensing and acting in that environment, and

3. the ***N-Jason***, (BDI-type) cognitive reasoning agent, which receives percepts from the virtual character, and generating plans delivered to the virtual character.

The sensory data from the virtual character is usually raw, so it must be converted into a symbolic representation to be used in the ***N-Jason*** agent (see 'Interpreter' in Virtual Agent block, a left of Figure 4-4). Likewise, the action plans from ***N-Jason*** agent are high level abstract behaviours that need to be decomposed in atomic virtual character actions, similarly (see the same, 'Interpreter' in Virtual Agent block, a left of Figure 4-4).

Percepts transferred from the virtual agent build up the belief set of the ***N-Jason*** reasoning agent, along with normative consequences (permissions, obligations, prohibitions) detached from the institution(s). The role of the institution(s) is in formulating and communicating normative consequences corresponding to environmental events delivered by ***N-Jason*** agents. As a part of belief set in the BDI reasoning agent, those normative consequences also contribute to the agent's decision-making process.

All decision making is performed by the ***N-Jason*** reasoning agent, each one of which is directly mapped to a virtual character. When we refer to 'Intelligent Virtual Agent (IVA)', we mean this pair of entities: the ***N-Jason*** reasoning agent and the virtual character.

The next section describes the interaction between the ***N-Jason*** agent and the institution in more detail.

4.4.2 Mental model

The mental model of the distributed agent framework is shown in Figure 4-5. As might be expected, this is somewhat different from a conventional agent platform, because

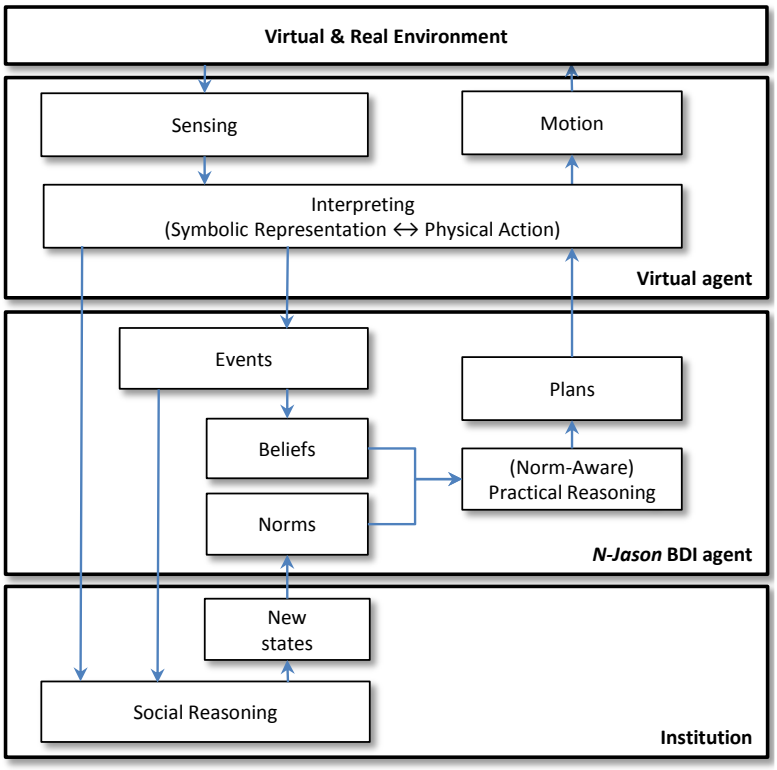

Figure 4-5: *Mental model of the distributed agent framework*

changes of mental state not only depend upon percepts from the VAs, but also on the normative information from the institution that are incorporated as percepts. In this section, we describe how this mental model was implemented, combining the institution, norms and the BDI agents. To begin with, definitions and formalizations are provided for the key concepts and entities. With these in place, we discuss how the mental state changes through interplay both between the institution and IVAs, and between the normative percepts (received from the institution) and mental states of IVAs.

Given an established VE, a group of IVAs co-exist and thus share a common environment, meaning they can each observe the same environmental changes, and can interact with each other.

Within the virtual world, the actions of the VAs bring about observable changes in the environment, which are perceived by the VAs and presented to the ***N-Jason*** agents and the institution as events. Thus, the VA act as sensors and interpreter for both BDI agent and institution and in consequence the agent and the institution are both aware of the same events (see left hand side of Figure 4-5). We refer to these as *external events*

because they occur external to the institution (see line 2 of Figure 4-2).

We use the concept of *event* to refer to the symbolic representation of some captured data about the environments. The *Recognising* component in the *VA* is responsible for turning whatever is observable in the VE into this symbolic representation. We assume that the perception and recognition of all external events is carried out by VAs. Although all these events are delivered to the ***N-Jason*** agent, only some of of them may be meaningful to the agent, and those that are not are ignored. The same applies to the institution: if there is no constitutive rule for an external event, then it is not of relevance for the institution. It should be noted that an event that is not meaningful in itself to the agent, but is to the institution, may consequently bring about an institutional change that is in turn relevant for the agent, because the event may cause the institutional recognition of a situation that affects the normative position of the agent: see the flow on the left hand side of Figure 4-5 from the institution to the agent. Thus, the institution consumes events from the VE and provides a social interpretation of them to the co-located IVAs, in terms of normative position information (permission, obligation), which is incorporated into the agent's mental model, broadly in line with the ideas set out in [Alechina et al., 2012], and subsequently may be taken into account by the agent's decision-making process. In so doing, the agents may exhibit socially aware responses to situations, which may be perceived by humans as more believable.

In the next section 4.4.2, we describe the details of such an interaction mechanism between the institution and IVAs inspired by on-line reasoning mechanism [Balke et al., 2011] in the institution. It is followed by the influence of social norms to the mental state in ***N-Jason*** agents of intelligent virtual agents in section 4.4.2.

Institutions and IVAs

Unlike the on-line reasoning Balke *et al.* propose, the instantiation of the institution is carried out by an external process called the *Institution Manager*, which provides the interface between the institution and the VA (event feed from VE) and the BDI agent[2]. The institution provides two services for the ***N-Jason*** agent:

[2]We decouple the institions and BDI-type cognitive agents using BSF whereas the institution and the agents are tightly coupled in the Balke *et al.*'s model. Due to the tightly coupled property, Balke *et al.*'s model is only able to instantiate the single institution. In contrast, the decoupling is the engineering choice in order to be able to instantiate multiple institutions simultaneously by the use of instituion manager.

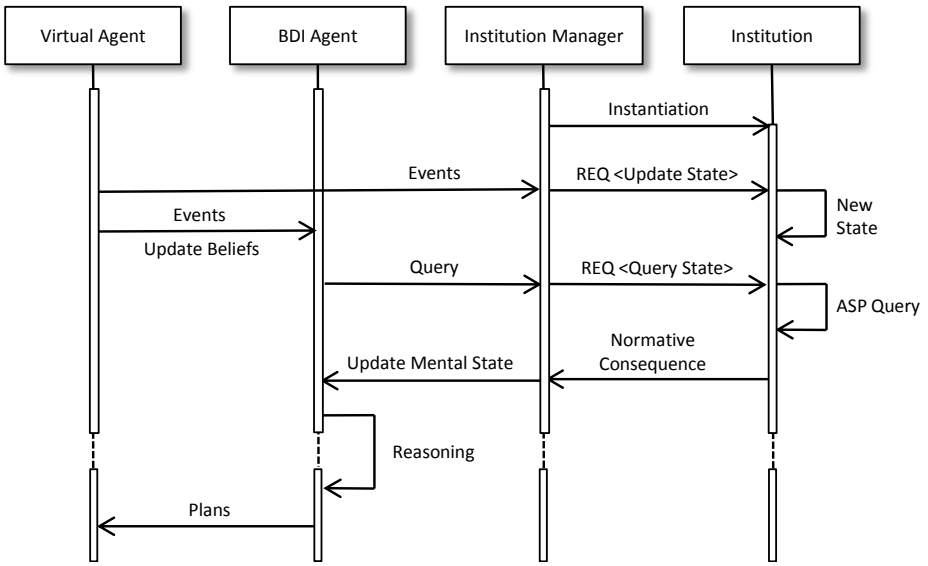

Figure 4-6: *Sequence Diagram of Runtime Reasoning*

- the interpreting of VE events and delivery of normative position updates, and
- a query mechanism, whereby the BDI agent can ask the institution for information about the normative position.

The sequence of the runtime reasoning model are depicted in Figure 4-6, reflecting the description above. Once the belief set in the ***N-Jason*** reasoning agent is updated by percepts (denoted as $\mathcal{E}_{ex}$) collected by the virtual character, then $\mathcal{E}_{ex}$ is delivered to the institution and the normative state ($\mathcal{P}$ or $\mathcal{O}$) is updated. When queries are made by the ***N-Jason*** reasoning agent to the institution, it gives rise to a social reasoning process and replies with new normative consequences($\mathcal{P}$ or $\mathcal{O}$) to the ***N-Jason*** reasoning agent. The mental state of BDI agent is updated by this social normative information, denoted as $\mathcal{N}(\mathcal{E}_n^{iva_i})$.

Thus, the IVA is able to extend the information it has thanks to the social reasoning capability provided by the institution. The notation $\mathcal{N}(\mathcal{E}_n^{iva_i})$ denotes the social normative information returned from the institution associated with the co-located IVAs. Once passed to the agent, the normative information can be incorporated in the belief base of the BDI agent.

Normative information and IVA Mental States

Normative information is communicated from the institution to the ***N-Jason*** agent and typically becomes part of the belief base. In this mental model, normative information ($\mathcal{N}(\mathcal{E}_n^{iva_i})$) is complementary information for each agent rather than a separate part of the mental state. This $\mathcal{N}(\mathcal{E}_n^{iva_i})$ becomes a part of the belief base and perhaps a subgoal for the achievement of primary goal. If an institutionally generated obligation is adopted as a final goal, the agent would in effect be regimented, thus the model can fall back to fully regimented, if the agents have such behaviours. The actual form of the normative information is represented as: $\texttt{obl}(\texttt{act}, \texttt{deadline}, \texttt{priority})$ or $\texttt{perm}(\texttt{act})$, which mean that an agent *X* is obliged to perform action *act* by deadline *deadline* with a priority *priority*, or an agent *X* is permitted to perform action *act*, respectively. A ***N-Jason*** agent will then take action *act*, if it satisfies a belief or a subgoal of its main goal. If not, the obligation is ignored and a sanction may follow. More details of decision making with norms in ***N-Jason*** are described in the next chapter (Chapter 5)

4.5 Illustrative Examples

We have set up two experimental scenarios in order to show how the system works. In the first we consider a queuing situation and in the second a situation concerning inter-personal distance.

We use *Second Life* as the VE in which all actions and interactions take place. IVAs are modelled by the combination of the *OpenMetaverse* library [OpenMetaverse Organization, 2012], for the virtual characters and BDI agents implemented in ***N-Jason***. *OpenMetaverse* allows not only for the creation of VAs, but also the scripting of behaviours using combinations of various atomic actions. ***N-Jason*** provides a platform for the creation and programming of norm-aware BDI agents using an extension of Agentspeak. *The institution* is specified using *InstAL*, the Institutional Action Language, which provides the social reasoning process.

4.5.1 Queuing

It is common in the society modelled in this scenario, that anyone who wishes to enter somewhere, or get on something is obliged to wait their turn in a queue. Of course,

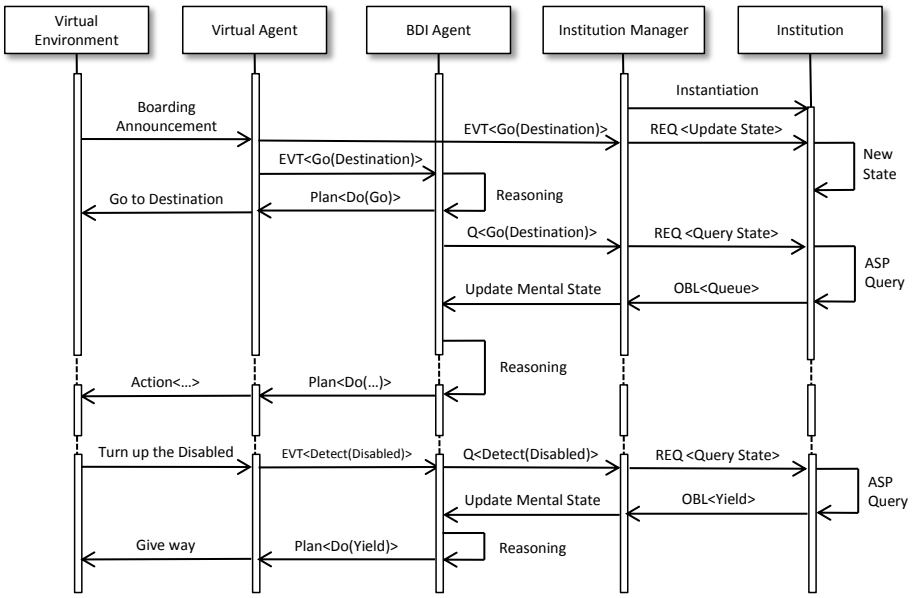

Figure 4-7: *Sequence Diagram of IVAs in Queue Scenario*

people who arrive later join the end of the queue. If someone arrives who is disabled or old, the social obligation is for those queuing to make a space for them at the front. Here, we model and demonstrate a such situation with two human users and three IVAs in a virtual world.

Initially, the VAs are dispersed, pursuing their own activities in the vicinity of the boat. As soon as the conductor (a human user) announces that boarding is starting, the VAs start to congregate at the gang plank and form a queue. When the disabled/old agent appears, the disabled/old agent requests them to make space for it. A sequence diagram for this scenario is provided in Figure 4-7.

In the normal course of events, IVAs take actions by individual reasoning, when beliefs are satisfied by its perceptions such as the case of *EVT<Go(Destination)>* (where *EVT*$< \cdots >$ denotes an event). In the mean time, the delivery of percepts may cause state update, or a query to the institution. Eventually however, the final decision on which action to take is accomplished through the combination of internal knowledge in BDI agents and normative information from the institution. Sometimes, such as in the case of *EVT<Detect(Disabled)>*, the outcome of social reasoning affects the agent behaviour directly. Nevertheless, it should be clear that plan selection is not purely determined by normative requirements, but depends on internal computation in *Jason*.

(a) Queuing (b) Detecting (c) Yielding

Figure 4-8: *Queuing*

Some frames extracted from the video are shown in Figure 4-8.

4.5.2 Inter-Personal Distance

Inter-Personal Distance(IPD), also known as proxemics, stands for the personal space between oneself and others [Hayduk, 1983]. We noted earlier, in Section 1.1, an interesting finding by Yee *et al.* that social interaction in VEs between avatars controlled by humans, appear to be subject to the same IPD social norms as in the real world. From this observational study, we may suggest that behaviours of VAs is likely to more believable if they are able to replicate human behaviours governed by social norms. Hence the motivation for the simulation of IPD in this second scenario.

To this end, we aim to show a simple IPD demonstration, in order to explore the effect of intimacy on IPD. The social norm in relation to intimacy and IPD is that: if people get too close to a person with whom they are not sufficiently intimate, then people typically either change their eye gaze or move to keep the proper level of IPD [Rosenfeld et al., 1984]. We simulate this with one human user and two IVAs.

The brief scenario for this example is that a human user meets two IVAs. One of the IVAs is friends with the human-controlled character, the other is not. Thus, when the human character greets them, the expected behaviour of the one known to the human is to react with greetings and maintain proximity. However, the other does not greet in return and also changes its eye gaze to maintain a proper level of IPD.

Figure 4-9 shows a sequence diagram of the IVAs used in the example. When one

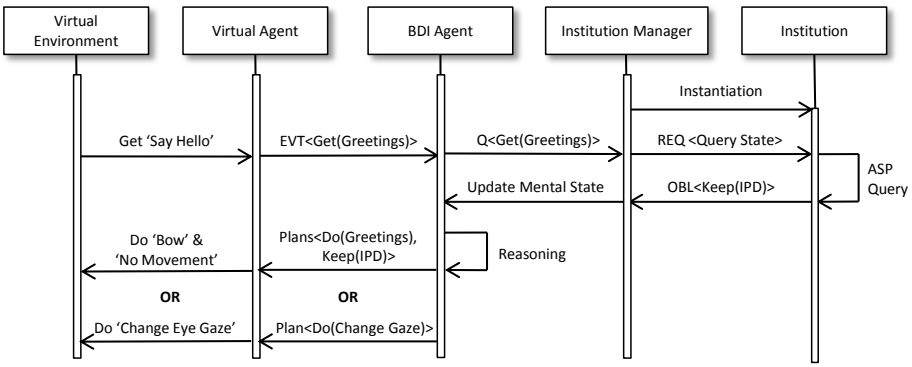

Figure 4-9: *Sequence Diagram of IVAs in IPD Scenario*

virtual agent observes an external greeting message (*'Say Hello'*), the greeting message is interpreted as an event (*EVT<Get(Greetings)>*) which in turn delivered to the ***N-Jason*** BDI agent. As soon as the percept of the ***N-Jason*** BDI agent is updated with the event (*EVT<Get(Greetings)>*), the query to the institution (*Q<Get(Greetings)>*) is made by the ***N-Jason*** BDI agent. Then the institution replies with a new normative consequence (*OBL<Keep(IPD)>*) to the ***N-Jason*** BDI agent after it's social reasoning process. The mental state of the ***N-Jason*** BDI agent is updated by this obligation afterwards, which leads the virtual agent's behaviour to comply with the obligation with the plans, either *Plans<Do(Greetings), Keep(IPD)>* or *Plan<Do(Change Gaze)>* depending on the IPD between the two virtual agents. Some frames extracted from the video are shown in Figure 4-10.

Figure 4-10: *IPD Model: Initial Situation (left), Different type of IPD behaviour (middle and right)*

4.6 Summary

In this chapter, what we mainly propose is a computational model, **D**istributed **N**orm-**A**ware **A**gent **A**rchitecture (**DNA**^{3}), in order to facilitate the socio-cognitive approach in reasoning virtual character behaviour with (social) norms.

To begin with, we present an explanation of socio-cognitive theory on human choices as a theoretical foundation of **DNA**^{3}, through the investigation on literatures which appear in the domain of psychology and social scinece. In here, the brief introduction to cognitive psychology is presented firstly. Then, criticisms on cognitive psychology is discussed as a motivation on the advent of socio-cognitive perspective on human decision making. Afterwards, we introduce the socio-cognitive theory itself in conjuction with the concept and mechanism of social cognition. In the end, the institution is investigated as a main element for social cognition process which produces norms, socially constructed knowledge, representing the correct behaviour in social context.

In accordance with the theoretical foundation, the computational model of the institution is presented as a means to social reasoning in conjuction with cognitive reasoning virtual characters. Then, this chapter introduces a **DNA**^{3}, which is established by the additional integration of institutions on top of coupling BDI-type cognitive agents and virtual characters. Given (**DNA**^{3}), we describe the informal explanation of the socio-cognitive reasoning process based upon run-time reasoning mechanism, and identified the mental model of ***N-Jason*** agent under the governance of the institution. Afterwards, two illustrative examples are given, which serve to illustrate how the VA, controlled by the ***N-Jason*** agent behaviours, is influenced by social reasoning in the institution.

When virtual characters are governed by the institution, their mental model is different from the conventional agents hence the decision making on norms and goals is essential. Since norms have its own properties such as deadlines, priorities and sanctions, the practical reasoning on norm compliance requires a careful consideration which can not be achieved in the conventional BDI-type agents. In the following chapter (Chapter 5), we introduce ***N-Jason*** and its internals, which is capable of run-time norm compliance via run-time norm execution mechanism on top of norm-aware deliberation.

Chapter 5

Norm Aware Decision Making in BDI Agents

As introduced earlier (Chapter 4), normative systems (institutions in this case) offer a means to govern agent behaviour in dynamic open environments. Under the governance, agents themselves must be able to reason about compliance with state- or event-based norms (or both) depending upon the formalism used.

This chapter describes how norm awareness enables a BDI agent to exhibit norm compliant behaviour at run-time taking into account normative factors. To this end, we propose ***N-Jason***, a run-time norm compliant BDI agent framework supporting norm-aware deliberation as well as run-time norm execution mechanism, through which new unknown norms are recognised and bring about the triggering of plans. To be able to process a norm such as an obligation, it is essential that the agent architecture is able to deal with deadlines and priorities, and choose among the plans triggered by a particular norm. Consequently, we extend the syntax and the scheduling algorithm of AgentSpeak(RT) [Vikhorev et al., 2011] to operate in the context of ***Jason***/AgentSpeak(L) and provide 'real-time agency', which we explain through a detailed examination of the operational semantics of a single reasoning cycle.

The chapter is organised as follows. We firstly start from the identification of the main objective in Section 5.1. In Section 5.2, we show an institutional context where ***N-Jason*** is running, specifically from the perspective of the semantics of norms. It is followed by Section 5.3, where we present a run-time norm compliant BDI agent framework including programming language and interpreter. After the operational se-

mantics in Section 5.4, related work and the contribution of this work are contrasted in Section 5.5. The summary and future work are discussed in Section 5.6.

5.1 Objective

In conventional development of BDI agents, norm compliance is typically achieved by design. That is, by specifying plans that are triggered by detached norms, because the agent programmer knows which norms the agent shall adopt, and then prioritising those rules so that the supporting norms are chosen over those preferred by the agent's mental attitudes, in order to suppress conflicts between the normative and the agent's existing goals. This creates an undesirable dependence between the agent implementation and the norm implementation, which creates two issues:

1. When an agent encounters new and unknown norms, which were not taken into account at design time, there is typically no plan to deal with those norms in the plan library at run-time. Hence, norm compliant behaviour cannot normally be exhibited because the norms are unavoidably ignored. Yet worse, agents may suffer a punishment from the enforcement of the normative system as a result of a violation caused by their incapacity to process the normative event.

2. The hierarchical prioritisation of normative over ordinary plans deprives an agent of its autonomy, since the norms in effect are treated as hard constraints, whose violation is not possible.

We believe that such tensions can be resolved by the use of an extended model of norm awareness. In the literature on BDI agents, norm awareness, which is a precursor to norm compliance, is typically manifested in two places:

1. at the *perception* level, by taking new unknown norms into account as part of the generic execution mechanism [Meneguzzi and Luck, 2009, van Riemsdijk et al., 2013] and

2. at the *deliberation* level, by attempts to resolve the conflict between normative factors and agents' mental attitudes [Alechina et al., 2012, Dybalova et al., 2013].

We propose to coalesce these approaches into one 'sense–think–act' reasoning cycle informed by the concept of awareness, which Charlton [Charlton, 2000] describes as the capacity *"to select and integrate relevant inputs from a complex environment to enable humans or animals to choose between a large repertoire of behavioural responses"*. This definition reminds us that, in order to be norm aware, agents should have knowledge (or understanding) about norms in respect of:

1. what (state) the norms are intended to reach or to achieve,
2. which action plans are appropriate to execute norms and
3. which behaviour agents should prefer between normative goals and the agent's own interests.

Thus, this chapter addresses the convergence of these approaches in the context of the BDI agent architecture, in order to be able to ground the discussion of how the extended model of norm awareness enables a BDI agent to exhibit norm compliant behaviour at run-time.

To do so, we propose ***N-Jason***, a run-time norm-compliant BDI agent framework supporting a run-time norm execution mechanism, under which new and unknown norms are recognised and enable the triggering of an appropriate plan (if present), in conjunction with norm-aware deliberation [Alechina et al., 2012].

In order to be able to process a norm such as an obligation, the agent architecture should be able to deal with deadlines and priorities, and choose among plans triggered by a particular norm. Consequently, we extend the syntax and the scheduling algorithm of AgentSpeak(RT) [Vikhorev et al., 2011] to operate in the context of ***Jason***/AgentSpeak(L) [Bordini et al., 2007] and provide 'real-time agency', which we explain through a detailed examination of the operational semantics of a single reasoning cycle.

5.2 Semantics of Norms

In this section, we briefly address the syntax and semantics of norms resulting from the institutional model introduced in Chapter 4. In addition, we sketch the concept of how

this set of norms can take part in the practical reasoning process of Norm-Aware BDI Agents.

Depending on the formalism of the normative system, norms can be categorised as state- or event-based. State-based norms usually express higher level norms that impose desirable or required states on the system (or an environment), often as a logical combination of institutional facts, which should be brought about by the actions of agents [Dignum, 2004]. In contrast, event-based norms generally represent relatively lower level activities addressing possibly executable events (or actions) at the individual agent level [De Vos et al., 2013]. As introduced earlier (Chapter 4), we use Cliffe's institutional model [Cliffe et al., 2007] for the purpose of providing detached event-based norms, upon which we develop the run-time norm compliance model presented here.

The institutional model is a set of rules describing consequences arising from observations for the purpose of reasoning about the current context, resulting in situation-specific norms. As an example, if an agent *X* is obliged to carry out an action *act* by deadline *deadline* otherwise the violation event *violation* is generated, the form of the normative information is represented as:

```
obl(act, deadline, violation)          (obligation)
```

Also if an agent *X* is permitted to perform an action *act*, then the representation is:

```
perm(act)                              (permission)
```

With regards to the norm compliance in BDI agents, van Riemsdijk *et al.* suggest in [van Riemsdijk et al., 2013] that one feasible approach for run-time norm execution is the use of *"pre-existing capabilities"* in the agent program when an agent encounters new and unknown norms. This assumes that event-based norms can identify the associated necessary actions, since event-based norms typically refer to relatively low-level activities that address possibly executable events (or actions) at the individual agent level [De Vos et al., 2013]. If appropriate information can be extracted from the detached norm, such that it is recognisable to an agent, in this way an agent presumably may execute unknown norms and so exhibit a form of norm compliance at run-time.

For example, the `act` term in an obligation represents a similar level of knowl-

edge to plans or events in a BDI agent program. If an agent can retrieve and recognise what action (or event) is required to be achieved, then it can trigger certain plans and attempt to carry out such behaviour even though the norm is not handled explicitly in the agent specification. With regard to the norm-aware reasoning, an agent may deduce a preference, if it is able to know the relative priorities, and critical impact or the deadline of normative factors by extracting `deadline` and `violation` information. This norm-aware reasoning may allow an agent to pursue its own preferences between its own goals, norms and sanctions, by measuring feasibility, as proposed by Alechina *et al.* [Alechina et al., 2012]. In this chapter, we only use obligations for such purpose, in order to focus on the essential aspects of the agent's internal reasoning process. Additionally, we consider the handling of prohibitions for the compatibility with other normative systems, however they are not explicit in the institution mechanism employed here.

5.3 The *N-Jason* BDI Agent Framework

In this section we outline ***N-Jason***, a norm aware BDI agent interpreter and its programming language for run-time norm compliant agent behaviour. In principle, it extends ***Jason***/AgentSpeak(L) syntactically, semantically and in the reasoning process of the interpreter. In practice, ***N-Jason*** is conceptually similar to AgentSpeak(RT) [Vikhorev et al., 2011], which is capable of dealing with deadlines and priorities and scheduling intentions with the aim of providing real-time agency. ***N-Jason*** is conceptually a superset of AgentSpeak(RT), to which it adds normative concepts (i.e. obligations, permissions, prohibitions, deadlines, priorities and durations) and norm aware deliberation.

We firstly examine work to date with regards to the programming language aspect. This is followed by an informal explanation of the ***N-Jason*** reasoning cycle. Subsequently, we show how the extended model of norm awareness in BDI agents is established by the combination of the run-time norm execution mechanism and norm-aware deliberation.

5.3.1 The *N-Jason* Agent Programming Language

A ***N-Jason*** agent consists of four main components: beliefs, goals, events and a set of plans. Beliefs and goals are identical to those in standard ***Jason***, while events and plans are extended. We now give a brief summary of the extended features of the basic elements in the agent specification. We take advantage of ***Jason***'s plan annotation mechanism to provide deadline, duration and priority information, so that each feature is simply a term, such as `deadline(X)`, `duration(Y)` or `priority(Z)`, where the parameters are (positive) integer literals. The interpretation of these annotations and examples are covered in the following.

Belief: A *belief* represents agent's information (e.g. initial states of an agent, internal knowledge established through the reasoning cycle) and its knowledge about the environments wherein agents are situated (e.g. percepts observed by agents, messages containing the information about other agents and norms delivered from normative frameworks). Typically, a belief is represented as a grounded atomic formula. The collection of beliefs is referred to as a belief base, which contains belief literals in the form of belief atoms and negations.

Goal: A *goal* is one of two basic types: an achievement goal or a test goal. The former are usually specified as predicates prefixed by the '!' operator. This specifies a certain state of the environment that the agent wants to achieve, which is indicated when the predicate associated with its achievement goal is true. The latter test goal, for which the prefix is the '?' operator, indicates that agents want to know whether the associated predicate is a true belief.

Event: An *event* is the main component for triggering agent's plans. In principle, changes in agent's mental attitudes (i.e. beliefs, goals and intentions) give rise to events. There are two types of events: one is an addition event denoted by '+', which means the addition of a belief or an achievement goal. The other is a deletion event denoted by '–', referring to a retraction of an element in the belief base.

As in ***Jason***, an addition event is categorised by a belief addition event denoted by '+' and a goal addition event jointly denoted by '+' and '!'. All external belief changes bring about belief addition events, so as to initiate the execution

of corresponding plans. In contrast, the goal addition event results from both internal and external changes in goals. In other words, explicit goals from the users or other agents result in a goal addition event, but also a goal addition event can be generated by internal operations affecting the agent's mental attitude, such as the execution of subgoals triggered in response to an external event.

Support for normative concepts is provided by an extension of the syntax for an event by the addition of *deadline* and *priority* information. The deadline is a real time value indicating a deadline by which an intention should be achieved. It is expressed in a some adequate unit of real world time. When the deadline is passed, it is no longer feasible to achieve an intention or to give a response with a belief change. The priority is a positive integer value that expresses the relative importance between the achievement of an intention and responding to changes in a belief. A larger value reflects a higher priority. Both can optionally be specified in the annotation (a list of terms in between square brackets "`[`" and "`]`") at the end of an event. For example the event:

Listing 5.1 : *deadline* and *priority* in an event

```
1 +!at(X, Y)[deadline(900), priority(10)]
```

specifies the goal adoption that an agent moves to the coordinate (X, Y), by the deadline 900, with priority 10. By default, the deadline is taken as infinity and the priority as zero. Note that the deadline and priority annotations do not play a part in unification at plan selection stage.

Plan: A *plan* is a sequence of actions (and subgoals) which is a means to achieve a (main) goal or a means to respond to changes in beliefs by agents. The plan typically consists of a head and a body, but sometimes an optional plan label, which defines an index, a name and other information, can be specified. The head is composed of a triggering event, which specifies an event for which the plan is to be used and a context specifying the condition which must be true for the plan to be a candidate for execution. The body is a series of actions and subgoals to achieve a main goal.

The plan is extended to support normative concepts. Given the three main elements (beliefs, goals and events), a duration is proposed in ***N-Jason***, specifically in order to enable assessment of the *feasibility* of the plan associated with the

deadline (see Section 5.3.4). The duration is a non-negative integer value representing a required time to execute the plan. In principle, the duration may be determined by the summation of an execution time of each external action in the plan body. For simplicity, we follow the assumption described in [Alechina et al., 2012], that the estimated time for each external action is fixed and already known. Like deadline and priority, a duration can be optionally specified in the plan label in the form of an annotation (a list of terms in between square brackets "[" and "]"). For example, the plan:

Listing 5.2 : *duration* in a plan

```
1 @plan[duration(50)]
2 +!at(X, Y) : req(ag)
3 <- move_toward(X, Y);
4    !ack(ag).
```

is triggered by the request from the agent *ag* to move to the coordinate (X, Y), and then to send back an acknowledgement to *ag*. The required (or estimated) execution time of the plan is 50.

5.3.2 The *N-Jason* Interpreter

The interpreter plays an important role in the operationalisation of agent programs. The agent's belief base, intentions and events are manipulated by the interpreter, and practical reasoning consisting of deliberation and means-ends reasoning is performed to achieve a goal or to respond to environmental changes.

As mentioned at the beginning of Section 5.3, we extend the ***Jason***/AgentSpeak(L) [Bordini et al., 2007] interpreter, by taking into consideration AgentSpeak(RT) [Vikhorev et al., 2011], in order to propose ***N-Jason***. ***Jason***/AgentSpeak(L) – consisting of programming language and its interpreter – is one of the popular/representative agent platforms in the agent oriented programming community. Initially, programming platforms for BDI agents have been proposed that emphasise either the formal basis (e.g. 2APL [Dastani, 2008], GOAL [Hindriks, 2008]), or industrial use (e.g. Jadex [Pokahr et al., 2005], JACK [Winikoff, 2005]), although all are based upon theoretical foundation of BDI agents and its programming language, AgentSpeak(L), as put forward in [Rao and Georgeff, 1995, Rao, 1996]. Unlike the variants of AgentSpeak(L) noted above, ***Jason***/AgentSpeak(L) is a compromise between those two territories. In addi-

tion, it provides further features which enable easier and more efficient programming on top of simple, elegant and intuitive programming language [Bordini and Hübner, 2010]. For this reason, we choose to extend ***Jason***/AgentSpeak(L) to realise the ***N-Jason*** interpreter.

However, as discussed in Section 5.1, ***Jason***/AgentSpeak(L), as a conventional BDI programming platform, is not capable of dealing either with properties of norms such as deadline, priorities and sanctions or the execution of unknown norms in the agent program. Thus, we take into account the addition of norm-aware deliberation [Alechina et al., 2012] via the *'real-time agency'* proposed in AgentSpeak(RT) [Vikhorev et al., 2011] on top of ***Jason***/AgentSpeak(L). Run-time norms execution is also proposed as another extension to ***Jason***/AgentSpeak(L) in order to support the operationalisation of unknown norms which are not built into the agent program.

Given the extended features of the ***N-Jason***, during a single reasoning cycle, run-time norm compliance is accomplished by an extended model of norm awareness that has three steps:

1. *Event Reconsideration*, to find out what the norm is intended to achieve or to reach,

2. *Option Reconsideration*, to identify which plan is the most appropriate in response to the norm,

3. *Intention Scheduling*, to confirm the decision about which behaviour agent would prefer between goals, norms and sanctions.

The interpreter code of ***N-Jason*** is shown in Algorithm 2. B is the belief base, E is the event base, G is a set of goals and I is a set of intentions of an agent. The function *create-tevent* encodes a percept as a triggering event and returns it. The function *add-event* updates the agent's event base with an event which is a pair of a triggering event and an intention. The function *update-belief* updates the agent's belief base with a percept p. The function *type* returns a type of p, either *obligation* or *prohibition*, if p is a norm. The function *edp* constructs a triggering event using the terms in the event-based norm, if the type of p is a norm (e.g. obligations). The functions EVENT- and OPTION-RECONSIDERATION accomplish the run-time norm execution mechanism described in Section 5.3.3. The main algorithm of the SCHEDULE function which carries out norm-aware intention scheduling is shown in Section 5.3.4. The internal

Algorithm 2 ***N-Jason*** Interpreter Reasoning Cycle

1: B :=B_0 /* B_0 are initial beliefs */
2: G :=G_0 /* G_0 are initial goals */
3: E :=$E \cup G$
4: P :=$P \cup N$ /* P are percepts and N are norms */
5: **for all** $p \in P$ **and** $p \notin B$ **do**
6: te_p = *create-tevent(p)*
7: $R_{te_p} := \{\pi\theta \mid \theta$ is a most general unifier (mgu) for te_p and plan $\pi\}$
8: **if** $R_{te_p} \neq \varnothing$ **then**
9: E :=*add-event*$(E,\ te_p)$
10: **else if** $R_{te_p} = \varnothing$ **and** $type(p) = (obl \mid proh)$ **then**
11: E :=EVENT-RECONSIDERATION(E, p)
12: **end if**
13: B :=*update-belief*$(B,\ p)$
14: **end for**
15: **for all** $\langle te, \tau\rangle \in E$ **do**
16: $O_{te} := \{\pi\theta \mid \theta$ is an applicable unifier for te and plan $\pi\}$
17: $(\pi\theta\theta' : \tau) := S_O(O_{te})$ where θ' is a context unifier for te and plan π
18: **if** $(\pi\theta\theta' : \tau) = nil$ **then**
19: $(\pi\theta\theta' : \tau)$:= OPTION-RECONSIDERATION(te)
20: **end if**
21: **if** $(\pi\theta\theta' : \tau) \neq nil$ **and** $\tau \notin I$ **then**
22: $I := I \cup (\pi\theta\theta' : \tau)$
23: **else if** $(\pi\theta\theta' : \tau) \neq nil$ **and** $\tau \in I$ **then**
24: $I := (I \backslash \tau) \cup$ push $((\pi\theta\theta' : \tau)\sigma, \tau)$ where σ is a mgu for $(\pi\theta\theta' : \tau)$ and τ
25: **else if** $(\pi\theta\theta' : \tau) = nil$ **and** $\tau \in I$ **then**
26: $I := (I \backslash \tau)$
27: **end if**
28: I :=SCHEDULE(I)
29: **if** $I \neq \varnothing$ **then**
30: I :=EXECUTE(I)
31: **end if**
32: **end for**

operation of the ***N-Jason*** interpreter is extended from [Vikhorev et al., 2011]. We use the same notations as in [Vikhorev et al., 2011] for consistency and comparability.

We now give an informal explanation of one reasoning cycle in the interpreter. At the start (lines 1–4), we assume that an agent perceives knowledge (P) from its environment and about its normative positions (N) (e.g. obligations) from one or more institutional frameworks. N is treated just like P, that is a form of percept at this stage,

by the interpreter (line 4).

The belief base (B) and the event base (E) are updated by P in the belief update process (*belief-update-function (buf)* more precisely) (see lines 6–13). This belief update involves the creation/addition of events in response to each new percept. Once a percept (p) is encoded as a triggering event (te_p) by the function *create-tevent*, the interpreter checks whether te_p has a set of relevant plans R_{te_p}[1] in the plan library Π. If R_{te_p} is retrieved, then E is updated with the event, a pair of te_p and its intention, by the function *add-event*. If no relevant plan is retrieved, te_p is ignored but B is updated in any case with p by the function *update-belief*.The same approach is taken for norms when the norms and its relevant plans are already specified in the agent program. Otherwise, the event reconsideration process (line 11) starts to find out what the norms are intended to achieve, as the first step in run-time norm execution.

Next, the interpreter starts the reasoning process in order to determine an applicable plan[2] in the selected set of applicable plans (O_{te}). The selection function S_O chooses a single option from O_{te} as a result of the unification of event and context. If S_O retrieves nothing (denoted by nil), then the interpreter follows exactly the same path as described above. The option reconsideration process (line 19) tries to find out which action plans are appropriate to execute unknown norms, as the second step in run-time norm execution. See lines 17–20.

If one single applicable plan is successfully retrieved by S_O, then the means-ends reasoning adds the applicable plan (π) as an intended means (IM) on top of an intention (I). If te of π is an internal event then π added in the existing I, otherwise a new I is created with π to be added in there (line 21–27). This is followed by the intention scheduling process which returns a *preference maximal set* of intentions in deadline order (line 28). Afterwards, one intention, selected by the intention selection function S_I, is finally executed (line 30). The details of the remainder are exactly the same as in [Bordini et al., 2007] or [Vikhorev et al., 2011].

[1] A relevant plan for a particular event is a plan whose triggering event matches the particular event. There can be many relevant plans for each triggering event in general [Bordini et al., 2007].

[2] An applicable plan is a candidate plan for execution, which has a context that evaluates to true given the agent's current beliefs [Bordini et al., 2007].

5.3.3 Run-Time Norm Execution

In Section 5.3.2, we explained that run-time norm execution is realised by two steps: (i) event reconsideration and (ii) option reconsideration. Prior to defining those reconsideration processes, we firstly define a property of the *executability* of norms at run-time. We say that a norm such as `obl(evt, deadline, violation)`, is *executable* at run-time iff:

1. $p \in P$ and $type(p) = (obligation \mid prohibition)$, where p is a percept, formed from a list of terms such as $term$(“,” $term)^*$, in a set of newly observed percepts P at run-time;

2. $te_p \notin E$, where te_p is a triggering event generated from the percept p, and E is an event base, which is a set of events $\{(te, \tau), (te', \tau'), \ldots\}$, where an event is a pair of a triggering event and an intention (te, τ);

3. $edp(p) \neq nil$ and $\{(te_{edp(p)}, \tau_{edp(p)})\} \cap E \neq \varnothing$, where $edp(p)$ is a function extracting the obliged event together with its deadline and priority from p, $te_{edp(p)}$ is a triggering event of the $edp(p)$, an event term in the norm, and $\tau_{edp(p)}$ is an intention of $te_{edp(p)}$ and

4. $R_{te_{edp(p)}} \neq \varnothing$, where $R_{te_{edp(p)}}$ is a set of relevant plans.

The executability determines the necessity for further reconsideration for the new and unknown norms. If those norms are judged executable at the perception stage, the event-reconsideration process starts for the addition of such norms to the event base as triggering events. Similarly, the executability also enables option-reconsideration in order to execute an applicable plan in relation to the triggering events derived from the norms.

Event Reconsideration aims to verify that a norm perceived at run-time is executable, even though no corresponding plan exists in the agent program. If an event extracted from a detached norm has a relevance to a certain set of plans, it thus has potential to trigger specific ones, and it is then concluded that the norm is executable. If the norm is proven to be executable, the interpreter adds the norm to the event base E as an achievement goal addition event. The procedure for event reconsideration is as follows (see Algorithm 3):

Algorithm 3 Event Reconsideration

Require: $P := P \cup N$
Require: $te_p = \textit{create-tevent}(p)$
1: **if** $p \in P$ **and** $type(p) = obligation$ **then**
2: $te_{edp(p)}$ = *create-tevent*(*edp*(p))
3: $R_{te_{edp(p)}} := \{\pi\theta \mid \theta$ is a mgu for $te_{edp(p)}$ and plan $\pi\}$
4: **if** $R_{te_{edp(p)}} \neq \varnothing$ **then**
5: $E :=$*add-event*(E, te_p)
6: **end if**
7: **else if** $p \in P$ **and** $type(p) = prohibition$ **then**
8: $\Xi :=$*add-prohibition*$(\Xi,$ *edp*$(p))$
9: **end if**

1. Extract the terms representing an obliged event, a deadline and its priority[3] from the obligation by the function *edp*, whose practical implementation may vary, depending on norm representations in various systems (line 2),

2. Construct a new triggering event (an achievement goal addition event in this case) from the combination of extracted terms (line 2),

3. Query the existence of a set of relevant plans with such a constructed triggering event (line 3),

4. Add such triggering event to E, if relevant plans are successfully retrieved (line 5) and

5. If the norm is a prohibition, then the extracted event is added into the prohibition base (Ξ) (line 7 - 8) and will be revisited at the norm deliberation stage [4].

For example, suppose there is a detached obligation `obl(at(X, Y), 1030, 10)`. If relevant plans are not found in the agent program (plan library of an agent, to be precise) in response to the obligation, the function *edp* firstly extracts the event (`at(X, Y)`), deadline (`1030`) and priority (`10`) from the obligation. Next, the interpreter constructs a new triggering event (an achievement goal addition event as described

[3]In principle, the last term is an event which arises when a violation occurs. This value normally indicates the criticality of such a violation. Higher values represents a higher priority.

[4]***N-Jason*** supports prohibitions as described above, and is therefore compatible with normative systems supporting prohibitions, but we note that the institutional model described in Section 5.2 does not have an explicit representation of prohibition, but only the absence of permission.

above) such as `+!at(X,Y)[deadline(1030), priority(10)]` using the extracted information. Subsequently, the interpreter queries the existence of relevant plans to S_R once again with a new triggering event, `+!at(X,Y)[deadline(1030), priority(10)]`. If the retrieval of relevant plans is successful, then the original event, `+!obl(at(X, Y), 1030, 10)`, is added to E.

One exceptional aspect in event-reconsideration is the addition of a deontic event te_p (which is a detached norm) instead of a normal event $te_{edp(p)}$ (which is a newly constructed triggering event) into the event base E. In so doing, we intend to distinguish norm-triggered intentions from ordinary intentions that normal events trigger, so as to facilitate norm-aware deliberation (see Section 5.3.4) in ***N-Jason***. In principle, ***Jason*** creates different intentions in response to different triggering events. Given this characteristic, both a deontic and a normal event create a deontic and a normal intention in ***N-Jason***, respectively. The intended means included in both intentions are identical since a deontic and a normal event trigger exactly the same plan in an agent program. However, the properties (e.g. deadline and priority) of each intention are different. The normal intention follows the original deadline and priority specified in the plan. In contrast, the deontic intention has different deadline and priority, which are inherited from those in the detached norm. As a result, these intentions are the main source of norm-aware deliberation. An agent is able to deliberate on norms and agent's private goals through the evaluation of the relative importance and urgency using norm-triggered (i.e. deontic) intentions and ordinary event-triggered (i.e. normal) intentions.

Suppose a plan whose label is `example`, is specified in an agent program:

Listing 5.3 : An Example Plan with *duration, deadline* and *priority*

```
1 @example[duration(50)]
2 +!at(X, Y)[deadline(1000), priority(5)]
3 <- move_toward(X, Y);
4    !ack(ag).
```

Assuming that a normal event triggering `example` is added to event base E. Then it creates a normal intention using a pair of normal event and its associated plan `plan_example`, whose deadline and priority are 1000 and 5, respectively. Later, a detached obligation `obl(at(X, Y), 1030, 10)` is received. Following Algorithm 3, the deontic event `+!obl(at(X, Y), 1030, 10)` is added to E, since a relevant plan `example` is found. Consequently a deontic intention is created using a pair of a deontic event and its associated plan `example`. Its deadline and priority are 1030 and

Algorithm 4 Option Reconsideration

Require: $\langle te_p, \tau \rangle \in E$ where te_p is an event and τ is an intention
Ensure: $\pi\theta\theta'$ where θ' is a context unifier for $te_{edp(p)}$ and plan π
1: **if** $type(p) = obligation$ **then**
2: $te_{edp(p)} =$ *create-tevent*$(edp(p))$
3: $R_{te_{edp(p)}} := \{\pi\theta \mid \theta$ is a mgu for $te_{edp(p)}$ and plan $\pi\}$
4: **if** $R_{te_{edp(p)}} \neq \varnothing$ **then**
5: $O_{te_{edp(p)}} := \{\pi\theta \mid \theta$ is an applicable unifier for $te_{edp(p)}$ and plan $\pi\}$
6: $\pi\theta\theta' := S_O(O_{te_{edp(p)}})$ where θ' is a context unifier for $te_{edp(p)}$ and plan π
7: **end if**
8: **end if**

10, respectively, which are different from those in the normal intention. Obviously, we have two intentions whose properties are different, although the intended means are absolutely same. Hence, ***N-Jason*** is able to carry out norm-aware deliberation on norms and the agent's own goals using those intentions. If ***N-Jason*** simply adds a normal event instead of a deontic event when an obligation is detached, then norm-aware deliberation may not be feasible since there must be only one normal intention.

Option-Reconsideration is a central element in the practical reasoning process whereas the event reconsideration happens at the perception stage. The main objective of option reconsideration is the determination of an applicable plan corresponding to the new and unknown norm – whose executability is already verified – and is thus added to E as an achievement goal addition event. If the applicable plan is chosen, then it will probably be used to enact a norm-compliant behaviour, unless it is infeasible as judged by intention scheduling (described in Section 5.3.4). The procedure is shown in Algorithm 4.

Like *Event-Reconsideration*, te_p is generated by a new and unknown norm that does not have any relevant plans R_{te_p} at this moment. Thus at the beginning of the option reconsideration, the interpreter carries out the same process for event reconsideration:

1. Extract the event term $edp(p)$ of the norm in order to retrieve relevant plans $R_{te_{edp(p)}}$ (as before), if the type of p is a norm (i.e. an obligation) (line 1 - 2),
2. Retrieve the relevant plans corresponding to the $te_{edp(p)}$ by the unification of an atomic-formula in a triggering event and each plan in an agent (line 3),
3. Determine a set of applicable plans with the constructed triggering event (line 5) and

4. Select a single applicable plan as an intended means to which to commit, through the extended unification of a triggering event, a plan and a context (line 6).

5.3.4 Norm Awareness in Deliberation

Norm awareness in the deliberation process is achieved by the scheduling of intentions with deadlines and priorities. We extend the algorithm proposed in [Vikhorev et al., 2011] with the consideration of prohibitions in order to establish a conflict-free *preference maximal set* of intentions. In effect, this is like [Alechina et al., 2012] which proposes a scheduling algorithm that brings about a *preference maximal set* of intentions, but that depends upon (N-)2APL's parallel execution of plans, whereas here the scheduling algorithm for (***N-)Jason*** has to take account of the single-threaded plan execution model in ***Jason***.

The scheduling algorithm is introduced in Algorithm 5. A set of candidate intentions $I_C = \{\tau, \tau', \dots\}$, which is sorted in descending order of a priority, is inserted into a scheduling process. If each intention is *feasible*, i.e. a plan on top of the intention can be executed before the deadline and is not prohibited by a set of prohibition $\Xi = \{\xi, \xi', \dots\}$, then the intention is added to the *preference maximal set* (Γ) whose criteria are defined as follows:

1. An intention is feasible *iff* the execution of the intention is completed before its deadline, that is, for τ,

$$ne(\tau) + et(\tau) - ex(\tau) \leq dl(\tau)$$

   where τ denotes an intention, $ne(\tau)$ is the time at which τ will next execute, $et(\tau)$ is the time required to execute τ, denoted in the plan label, $ex(\tau)$ is the elapsed time to execute τ to this point, and $dl(\tau)$ is the deadline for τ specified in the plan [Alechina et al., 2012].

2. The intention should not be prohibited, that is, for τ

   - $\tau \notin \Xi$ or
   - $\tau \in \Xi$, then $\forall \xi \in \Xi$, $\tau = \xi$ and $priority(\tau) > max\{$*priority*$(\xi), \forall \xi \in \Xi\}$

Algorithm 5 Scheduling of Intentions

1: $\Gamma := \varnothing, \Xi' := \varnothing$
2: **for all** $\tau \in I$ in descending order of priority **do**
3: **if** $\{\tau\} \cup \Gamma$ is feasible **then**
4: **if** $\tau \notin \Xi$ **then**
5: $\Gamma := \{\tau\} \cup \Gamma$
6: **else**
7: **for all** $\xi \in \Xi$ **do**
8: $\Xi' := \{\tau\theta \mid \theta$ is a mgu for ξ and intention $\tau\}$
9: **end for**
10: **if** *priority*$(\tau) > max\{$*priority*$(\xi), \forall \xi \in \Xi'\}$ **then**
11: $\Gamma := \{\tau\} \cup \Gamma$
12: **end if**
13: **end if**
14: **end if**
15: **end for**
16: sort Γ in order of increasing deadline
17: **return** Γ

where τ is an intention, ξ is a prohibited event in the prohibition base Ξ and *priority* is a priority retrieval function.

Scheduling in ***N-Jason*** is also pre-emptive in that the adoption of a new intention τ may prevent scheduled intentions with lower priority than τ (including currently executing intentions) being added to the new schedule just as in N-2APL and AgentSpeak(RT). Intentions that cannot meet their deadline are dropped.

5.3.5 Implementation

We have implemented ***N-Jason*** on top of the existing code base for ***Jason*** version 1.3.6. The latest prototype[5] of ***N-Jason*** implements the core language extensions (i.e. syntax, semantics) described in Section 5.3.1 and the extensions (e.g. run-time norm execution, norm-aware deliberation) described in Section 5.3.3 and Section 5.3.4. In addition, we implement a norm adoption mechanism in ***N-Jason***, so that an agent under the governance of institutional frameworks is able to receive situationally appropriate norms and subsequently add them as percepts for processing by the reasoning cycle.

[5] ***N-Jason*** is available via `http://bsf.googlecode.com/svn/tags/njason-0.0.1/`

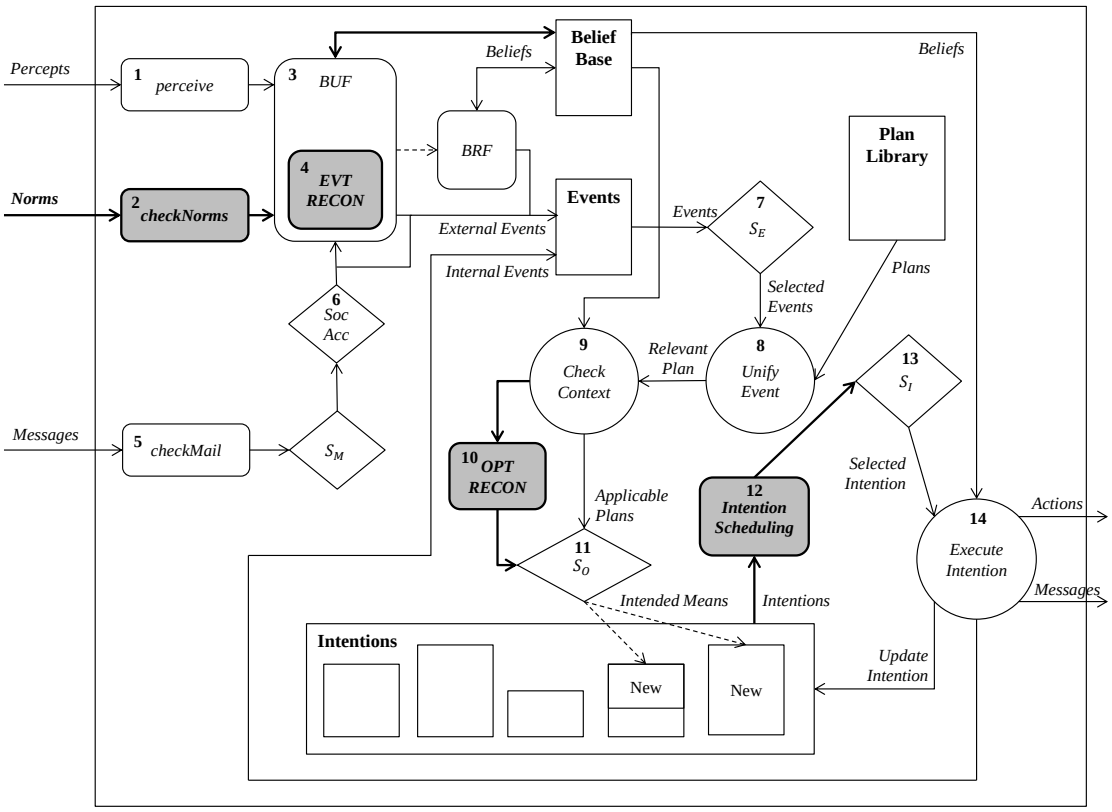

Figure 5-1: *Extended Features of* **N-Jason** *on* **Jason**/*AgentSpeak(L)*

Figure 5-1 shows the extended features[6] and how they fit into the ***Jason*** interpreter. The language extensions, run-time norm execution and norm-aware deliberation are implemented as part of the reasoning cycle of the ***Jason*** interpreter. The norm adoption mechanism is implemented as an extension of the `AgArch` class.

In brief, one reasoning cycle of ***Jason*** is modelled as a transition system over states. The configuration of a ***Jason*** agent [Bordini et al., 2007], contains the current state, denoted s, where $s \in$ {ProcMsg, SelEv, RelPl, ApplPl, AddIM, SelInt, ExecInt, ClrInt}. Each state has a corresponding procedure – `applyProcMsg()`, `applySelEv()`, `applyRelPl()`, `applySelAppl()`, `applyFindOp()`, `applyAddIM()`, `applyProcAct()`, `applySelInt()`, `applyExecInt()` – which are internal to the `reasoningCycle()` method in the transition system. To this transition system, we add the states RcvNorm and

[6]Grey boxes with numbers 2, 4, 10, 12 in Figure 5-1 are new features. For more explanation about other white boxes see Chapter 4 of [Bordini et al., 2007].

`schInt` and customise the procedures `reasoningCycle()`, `applyRelPl()`, `applyFindOp()`, `applySelInt()`. The complete states are detailed in Section 5.4.1.

We now sketch some details of the implementation. To begin with, we extend the ***Jason*** agent reasoning architecture class (`AgArch`) by subclassing in order to facilitate the norm adoption mechanism ($^{2}checkNorms$). Run-time norm execution is achieved by *Event-* and *Option-Reconsideration* as described in Algorithms 3 and 4 above. The former, $^{4}EVTRECON$ in Figure 5-1, is implemented by customising the native belief update function (`buf()`) which is a subroutine of `reasoningCycle()` to implement Algorithm 3. For the latter, $^{10}OPTRECON$ in Figure 5-1, we customise the native option selection function (`applyFindOp()`) to implement Algorithm 4. Norm-aware deliberation, $^{12}IntentionScheduling$ in Figure 5-1, is accomplished by an intention scheduler with deadlines and priorities implemented in a newly created `IntentionScheduler` class. The scheduling (`schedule()`) method in this class is inserted just before the intention selection function (`selectIntention()`) which is a part of the `applySelInt()` procedure.

Apart from the above changes to the interpreter, the language syntax extensions are implemented by the customisation of the annotation processing routine (`setLabel()` and `setTEvent()`) in the `Plan` class.

5.3.6 Example

As an example, we consider robots serving beer in a pub, whose main role is to get an order and to deliver a beer to the customer. We assume the existence of some institutions delivering desirable social norms, subject to the observations of participants, and that all agents are governed by such systems. A part of the agent program is shown below:

Listing 5.4 : A Part of a *N-Jason* Example Program

```
1 @P1[duration(5)]
2 +!at(X, Y) : not at(X, Y)
3 <- moveToward(X, Y).
4
5 @P2[duration(10)]
6 +!order(X, Y)
7 <- get(beer);
8    moveToward(X, Y).
9
```

```
10 // A request from customer seated at (X, Y).
11 // The deadline is D and the priority is P.
12 +request(X, Y)[deadline(D), priority(P)]
13 <- !order(X, Y)[deadline(D), priority(P)].
```

At time 100, the robot receives the following events:

E1: `+!request(2, 3)[deadline(130), priority(20)]`
A request from customer seated at (2, 3).
The deadline is 130 and the customer is important so the priority is 20.

E2: `+!request(1, 1)[deadline(115), priority(10)]`
A request from customer seated at (1, 1).
The deadline is 115 and the the priority is 10.

E3: `+!request(3, 3)[deadline(130), priority(10)]`
A request from customer seated at (3, 3).
The deadline is 130 and the the priority is 10.

These three events trigger the plan P2, and give rise to three possible intentions τ_1 (P2 triggered by (2, 3)), τ_2 (P2 triggered by (1, 1)) and τ_3 (P2 triggered by (3, 3)). τ_2 is not feasible, thus it is dropped, whereas τ_1 and τ_3 are feasible, so scheduled in deadline order: τ_1 is scheduled first between 100 and 110 since it has an earlier deadline followed by τ_3 between 110 and 120. Now the agent starts the execution of τ_1.

Now consider an announcement of a fire alarm by one of the normative frameworks. It broadcasts an obligation containing the coordinates of an exit to all participants so they may escape from the building. Suppose the norm is `obl(at(0, 0), 115, 100)`. Although the obligation is not stated in the agent's program, it is executable since the agent has a *pre-existing* moving ability `!at(X, Y)`, which is enough to satisfy the obligation. With the event- and option-reconsideration, the event :

E4: `+!at(0, 0)[deadline(115), priority(100)]` is generated from the obligation, thus adopting the plan P1, bringing about an intention τ_4 (P1 triggered by (0, 0)). During the execution of τ_1, τ_3 and τ_4 are inserted into a new schedule in deadline order: since the priority of τ_4 is greater than τ_3 and τ_4 has a more urgent deadline, the agent starts to execute τ_4, triggered by the obligation, before the execution of τ_3.

Notwithstanding, that this example is extremely simple, it provides a useful in-principle illustration of norm-aware deliberation – as performed by intention scheduling – as well as the run-time norm execution mechanism in ***N-Jason***.

5.4 Operational Semantics

In this section, we present a theoretical foundation for the ***N-Jason*** programming language with semantics based upon an extension of the operational semantics for ***Jason***/AgentSpeak(L). Given the formal semantics of ***Jason*** we extend the transition rules which transform one extended configuration into another. To begin with, we show a configuration of individual ***N-Jason*** agents which is almost unchanged except for norm configuration. In the following section, we describe the transition rules that give rise to a configuration change at each state in a single reasoning cycle. For consistency and comparability, we follow exactly the same notations as those in published ***Jason*** descriptions excepting the normative aspects.

5.4.1 *N-Jason* Configuration

The configuration of ***N-Jason*** is a tuple $\langle ag, C, N, T, s\rangle$ where:

- ag is an agent program consisting of a set of beliefs bs and a set of plans ps, as defined by the EBNF in [Bordini et al., 2007].
- An agent's circumstance C is a tuple $\langle I, E, A\rangle$, where I is a set of *intention* $\{i, i', \ldots\}$, E is a set of *events* $\{(te, i), (te', i'), \ldots\}$, in which event is a pair of a triggering event and an intention (te, i) and A is a set of actions an agent performs in the external environment.
- N is a tuple $\langle \Gamma, \Xi\rangle$ denoting normative consequences delivered from normative systems, where Γ is a set of obligations $\{\gamma, \gamma', \ldots\}$ and Ξ is a set of prohibition $\{\xi, \xi', \ldots\}$.
- T is a tuple $\langle R, A_p, \iota, \varepsilon, \rho\ \rangle$ defining a trace of provisional information required for subsequent steps within a single reasoning cycle, where R is the set of *relevant plans*, A_p the sets of applicable plans, and ι, ε and ρ record an intention,

event, and applicable plan (respectively) at a specific moment under consideration within the execution of a single reasoning cycle.

- The current state s within an agent's reasoning cycle is denoted by $s \in \{$RcvNorm, ProcMsg, SelEv, RelPl, ApplPl, AddIM, SchInt, SelInt, ExecInt, ClrInt$\}$.

5.4.2 Transition Rules

The execution of the ***N-Jason*** program leads the modification of the initial configuration of an agent via transition rules given below. We do not repeat the communication semantics, since these are unaffected by the changes in relation to norms.

In general, the transition would normally start from the state ProcMsg, but we propose a preceding step RcvNorm, as described in Section 5.2 because this provides the hook for the consideration of the norm as part of the reasoning cycle. Thus, note that the initial configuration of this model is $\langle ag, C, N, T,$ RcvNorm$\rangle$, where ag is specified by the agent program and other all components are empty, and the reasoning cycle starts from RcvNorm with the transition rules given below.

Receiving detached norms: As described in Section 5.2, institutional frameworks may distribute norms via broadcasting when a norm is activated by the fulfilment of institutional states triggered by external events in the environment. As soon as the event-based norms are received, the norms effectively act like an ordinary event thus triggering the transition of the agent's mental state. Rule $\mathbf{RcvNorm}$ (see Figure 5-2) updates the agent belief base and an event base component C_E associated with adding new norms, specifically in the case of obligations in an obligation base N_Γ. Otherwise, only a prohibition is added into the prohibition base and there are no updates to other components.

Relevant plans: (see Figure 5-3) If the transition of states (RcvNorm $\mapsto$ SelEv) is successful after RcvNorm and the state SelEv selects one event from the component E of which event is either $\langle te, i\rangle$ or $\langle \gamma, i\rangle$, rule $\mathbf{Rel_1}$ starts to assign the set of relevant plans to component T_R in the state RelPl. Rule $\mathbf{Rel_2}$ indicates the reconsideration situation where a new triggering event extracted from the obligation is assigned to the component C_E, where Evt(γ) is a function constructing a triggering event by the retrieval of information from γ. Rule $\mathbf{Rel_3}$ assigns a set of relevant plans to T_R in respect of the reconsidered event. Rule $\mathbf{Rel_4}$ and $\mathbf{Rel_5}$ cope with the situation where no rel-

$$\frac{N \neq \{\}}{\langle ag, C, N, T, \mathsf{RcvNorm}\rangle \rightarrow \langle ag', C, N', T, \mathsf{SelEv}\rangle} \quad \textbf{(RcvNorm)}$$

where: $ag'_{bs} = ag_{bs} \cup \{\gamma\}$
$N'_\Gamma = N_\Gamma \cup \{\gamma\} \vee N'_\Xi = N_\Xi \cup \{\xi\}$

Figure 5-2: *Transition Rule for Receiving a Norm*

$$\frac{T_\varepsilon = \langle te, i\rangle \quad \mathsf{RelPlans}(ag_{ps}, te) \neq \{\}}{\langle ag, C, N, T, \mathsf{RelPl}\rangle \rightarrow \langle ag, C, N, T', \mathsf{ApplPl}\rangle} \quad \textbf{(Rel}_\textbf{1}\textbf{)}$$

where: $T'_R = \mathsf{RelPlans}(ag_{ps}, te)$

$$\frac{T_\varepsilon = \langle \gamma, i\rangle \quad \mathsf{RelPlans}(ag_{ps}, \gamma) = \{\}}{\langle ag, C, N, T, \mathsf{RelPl}\rangle \rightarrow \langle ag, C', N, T, \mathsf{RelPl}\rangle} \quad \textbf{(Rel}_\textbf{2}\textbf{)}$$

where: $C'_E = \{\langle \mathsf{Evt}(\gamma), i\rangle\}$

$$\frac{T_\varepsilon = \langle \mathsf{Evt}(\gamma), i\rangle \mathsf{RelPlans}(ag_{ps}, \mathsf{Evt}(\gamma)) \neq \{\}}{\langle ag, C, N, T, \mathsf{RelPl}\rangle \rightarrow \langle ag, C, N, T', \mathsf{ApplPl}\rangle} \quad \textbf{(Rel}_\textbf{3}\textbf{)}$$

where: $T'_R = \mathsf{RelPlans}(ag_{ps}, \mathsf{Evt}(\gamma))$

$$\frac{\mathsf{RelPlans}(ag_{ps}, te) = \{\}}{\langle ag, C, N, T, \mathsf{RelPl}\rangle \rightarrow \langle ag, C, N, T, \mathsf{SelEv}\rangle} \quad \textbf{(Rel}_\textbf{4}\textbf{)}$$

$$\frac{\mathsf{RelPlans}(ag_{ps}, \mathsf{Evt}(\gamma)) = \{\}}{\langle ag, C, N, T, \mathsf{RelPl}\rangle \rightarrow \langle ag, C, N, T, \mathsf{SelEv}\rangle} \quad \textbf{(Rel}_\textbf{5}\textbf{)}$$

Figure 5-3: *Transition Rules for Relevant Plans*

$$\frac{T_\rho = \{\}}{\langle ag, C, N, T, \mathsf{SchInt}\rangle \rightarrow \langle ag, C', N, T, \mathsf{SelInt}\rangle} \quad \textbf{(SchInt)}$$

where: $C'_I = \text{SCHEDULE}(C_I)$

Figure 5-4: *Transition Rule for Scheduling Intentions*

evant plan is retrieved. In those cases, events (both ordinary event and reconsidered event) are simply ignored and the state returns to SelEv.

Since transition rules between (AppPl $\mapsto$ AddIM) are almost same as those in ***Jason*** we give a brief description of each rule at each state from here. If T'_R is successfully assigned then it is followed by: (i) AppPl which assigns a set of applicable plans to T_{A_P} by retrieving those relevant plans whose contexts are believed to be true, (ii) SelAppl which assigns a particular intended means selected by an option selection function S_O

to T_ρ, and (iii) AddIM which adds a selected intended means to C_I which is an existing intention or a newly created one. If transitions fail between (AppPl $\mapsto$ AddIM), then the state SelInt becomes the next step. For more information, see [Bordini et al., 2007].

Scheduling of intentions: Rule **SchInt** (see Figure 5-4) updates the component C_I' by the function SCHEDULE(C_I). Note that the scheduling function, SCHEDULE(C_I), sorts intentions in order of priority and deadline so as to determine the *preference maximal set* of intentions discussed in Section 5.3.4.

After this step, the transition system follows the same rules as presented in [Bordini et al., 2007] in order to execute an intended means in an particular intention selected by S_I in between SelInt, ExecInt and ClrInt.

5.5 Related Works

There has been much research over a number of years on the matter of norm compliance through the combination of normative frameworks and classical (BDI-type) cognitive agents [Boissier et al., 2013, Hubner et al., 2007]. However, research on compliance of norms at the individual agent level has received less attention. As discussed in Section 5.1, this problem can be decomposed into two perspectives: to facilitate a generic norm execution mechanism at run-time, and to focus on the process of rational decision making between norms and existing goals.

Alechina *et al.* [Alechina et al., 2012] introduce N-2APL, a norm-aware BDI agent architecture and its programming language. It is able to carry out norm-aware deliberation, which aims to permit agents to resolve the conflicts between an agent's own goals, normative goals and sanctions. This is accomplished by a deadline- and priority-based intention scheduling algorithm, which weighs the feasibility for all intentions that may bring about conflicts. The (potential) sanctions may affect agent decision making, but violations are possible in this approach. Given N-2APL, Dybalova *et al.* [Dybalova et al., 2013] demonstrate norm-compliant agents in location-based gaming environments in conjunction with the organisational framework, 2OPL [Dastani et al., 2009]. There, once organisations have broadcast state-based norms to all participants, the individual agents achieve a state of the environment described in the norms using a design-based approach. ***N-Jason*** is also able to support norm-aware deliberation in conjunction with an institutional model, which is similar to the combination of N-2APL and

2OPL, but extends the concept of norm awareness to the whole reasoning cycle. As a result, it supports agents in being design-based norm compliant, but can additionally deliver run-time compliance through norm execution.

Meneguzzi *et al.* [Meneguzzi and Luck, 2009] focuses on norm awareness at the perception level, by extending the AgentSpeak(L) BDI architecture with a run-time plan modification technique. It enables agents to behave appropriately in response to newly accepted norms at run-time. However, it assumes that the norms are non-conflicting, so it does not consider scheduling of plans with regards to their deadlines or possible sanctions in accordance with existing goals in agents. Whereas [Meneguzzi and Luck, 2009] takes a rather practical perspective, van Riemsdijk *et al.* [van Riemsdijk et al., 2013] introduce a formal framework for generic norm execution, which allows agents to be norm compliant by triggering or preventing actions in new and unknown norms at design time. However the agent in [van Riemsdijk et al., 2013] works at the level of individual actions (its decision mechanism chooses actions rather than plans) and the norms are specified in terms of actions, making in effect a norm-reactive agent, and it is unclear how the decision mechanism can combine actions to achieve goals and thereby the objective of a norm-deliberative agent. In ***N-Jason***, run-time norm execution is in practice accomplished at the level of plans to achieve goals, and norms indicate a sort of event that triggers plans. Moreover, in ***N-Jason*** run-time norm compliance is achieved on top of the norm aware decision making and in conjunction with the execution mechanism.

Notwithstanding the benefits of ***N-Jason***, there are some issues to highlight in respect of the mechanism for run-time norms. The norm compliance strategy is hard-coded in the semantics of the language, leaving only a capacity for configuration via the plan annotations, whereas the strategy is programmable through agent plans (i.e. supporting the design of strategy by an agent programmer) in JaCaMo [Boissier et al., 2013] and N-2APL [Alechina et al., 2012]. Thus, the proposal presented here provides a pre-packaged approach to normative reasoning, since it deprives the agent of the scope to change plans dynamically or mis-behave intentionally, based on rules the agent programmer designs. However, the mechanism put forward here does enable legacy agents, which have no compliance rule or strategy in their specification, to become norm-aware automatically. Thus, those agents' behaviour can be coordinated through the governance of normative frameworks without further engineering effort.

Another issue lies in the simple mechanism for the operationalisation of norms in run-time norm execution. The approach described here means the ontology and syntax of norms that can be executed are limited to those present in the plan library of an agent. In consequence, some detached norms, that may correspond semantically to one of an agent's plans, but which are ontologically different from the plan, will be ignored or violated. To this end, it would be worth to generalise the execution mechanism with the analysis of semantics of norms, following [van Riemsdijk et al., 2013], in conjunction with plan synthesis.

5.6 Summary

In this chapter, we have presented a design for a norm-aware BDI agent, ***N-Jason***, that enables the exhibition of norm compliance at run-time. Basically ***N-Jason*** offers a generic norm execution mechanism on top of norm-aware deliberation to contribute to the exploitation of run-time norm compliance. Run-time norm execution specifically focuses on the operationalisation of new and unknown (event-based) norms not stated in the agent program at run-time. By judging the executability of them, ***N-Jason*** agents executes those norms following an extended model of norm awareness consisting of: (i) *event reconsideration*, to find out what the norm is intended to achieve or to reach, and (ii) *option reconsideration*, to identify which plan is the most appropriate in response to the norm. The selection of norm compliant behaviour is achieved in the norm-aware deliberation process by *intention scheduling* with deadlines, priorities and prohibitions which confirms the decision about which behaviour agent would prefer between goals, norms and sanctions. It brings about a *preference maximal set* of intentions in order to realise the norm compliance. ***N-Jason*** is implemented in ***Jason***/AgentSpeak(L) and extends its syntax and semantics to create ***N-Jason***.

We believe that run-time norm compliance model is beneficial for the enhancement of both a norm compliance capability and agent autonomy from the agent's perspective. However, we note that the behaviour triggered by run-time norm execution may look like unpredictable/unwanted behaviour from the agent programmer's perspective.

Chapter 6

Case Studies

Through the previous chapters (Chapter 3, 4 and 5), we introduce **DNA**^{3} (which consists of normative framework, norm-aware ***N-Jason*** agent and virtual characters) and its internal mechanism grounded on socio-cognitive theory in order to design and simulate virtual character behaviour with norms. In this chapter, we present three case studies to show the adequacy of **DNA**^{3} in engineering norm-aware behaviour of virtual characters.

The use of polite agents is a new approach in order to improve the navigation experience for player characters (PCs) in crowded virtual worlds [Allen et al., 2012]. This chapter aims to demonstrate the politeness of virtual characters using **DNA**^{3}, subject to a theory of politeness composed of conventional and interpersonal politeness.

According to status-power theory by Kemper [Kemper, 2011], the choice of social behaviour is strongly influenced by the social status as well as the social force that the current social context imposes. In this sense, the form of polite behaviour could vary depending on the social situations, i.e. whenever the situation is changed, then the social norms and its associated polite behaviour should be also changed to show the appropriateness of politeness in that new situation. Given this hypothesis, we present three types of polite behaviour in relation to three different situations.

We firstly present the polite behaviour on the individual level, avoiding the disturbance of others in navigation, inspired by Allen *et al.* [Allen et al., 2012]. Afterwards, we extend this polite behaviour in navigation when individuals are formed into a group (which gives rise to the change of social status). Taking into account the human territoriality model [Scheflen and Ashcraft, 1976] as a way to propose group formation subject to social relationship, we first show here the formation of groups subject to so-

cial norms which the model proposes, then the polite behaviour (avoiding disturbance in this case) between those groups is described. In addition, the evacuation model is illustrated when the situation is rapidly changed from ordinary group level navigation to the emergency situation.

Through these case studies, we aim to show how the most appropriate (polite) behaviour can be determined by incorporating social and cognitive reasoning so as to ensure the adequacy of **DNA**^{3} for modelling norm-aware virtual characters in all circumstances with regards to (i) the capability in reasoning about social norms subject to changes of social situations and (ii) the rationality in decision making on norms and agent's individual goals.

6.1 Intro: Politeness in Virtual Characters' Behaviour

6.1.1 Motivation

Navigation in virtual environments (VEs) is an essential task for virtual humans, but it remains challenging for several reasons. A significant contributing factor is that VEs are getting more sophisticated, dynamic, and crowded, due to the outstanding advances in recent years resulting in changes in form, scope and purpose [Messinger et al., 2009]. In addition, the awkwardness of many user input devices severely impacts on ease of navigation of player characters (PCs).

These circumstances lead to a poor interaction experience for users, with both satisfaction and believability likely to be decreased [Merritt et al., 2011] in consequence. In this context, the use of politeness in virtual agents is proposed as an approach for a resolution of such tensions. Allen *et al.* [Allen et al., 2012] put forward the idea that polite behaviours of non-player characters (NPCs) may promise a more pleasant navigation experience for PCs in virtual worlds. Allen uses a predictive model to understand a PC's intention, which allows virtual humans to behave politely (collision avoidance in this case) and so improves the interaction in dense crowds.

Presumably, politeness looks like an aspect of social intelligence since such polite behaviours are brought about by prediction, accompanied by recognition of other's behaviour, in social relationships. This view is supported by the literature on the theory of politeness. A fairly recent and popular theory of politeness is presented in [Arndt

and Janney, 1985]. Traditionally, the notion of politeness has been discussed in the context of linguistic theory [Brown and Levinson, 1987], but Arndt *et al.* extends the discussion in terms of verbal and non-verbal communication. They decompose politeness into *conventional* politeness and *interpersonal* politeness. The former refers to compliance with social conventions (or etiquette) that maintains order in society. The latter, interpersonal politeness, refers to the considerations of others and their feelings in social interactions. This implies that more comprehensive awareness in social situations (beyond conversational situations) ought to be required to be polite in some personal situations. Accordingly, it is clear that politeness seems to be a part of, or closely related to social intelligence, which enhances the adequacy of human behaviour to society or its members in public and (or) private social situations.

The use of norms, normative frameworks and norm-aware virtual characters is one potentially effective approach to satisfy the above theory. In effect, normative frameworks provide a repository of knowledge of social conventions, particularly about correct behaviours, by capturing human social structures [Boella et al., 2006]. In addition, such a framework may also infer expected behaviours, subject to specific situations, by means of logic-based reasoning processes. It seeks to identify the causality between observations in the external world and potentially correct behaviours that are only meaningful within given social situations [Cliffe, 2007]. This can be viewed as a sort of prediction about rational decisions corresponding to the specific interpersonal context.

Hence, it seems clear that modelling behaviour, subject to the theory of politeness, is feasible with assistance from normative frameworks. Virtual humans may be able to behave adequately under the governance of social rules – conventional politeness, but may also take actions as a result of recognizing situation-specific actions from observing the activities of others associated with personal interactions – interpersonal politeness.

In this context, this chapter aims to demonstrate the enhancement of politeness using norm-aware virtual agents which is engineered by **DNA**^{3}, as the combination of normative frameworks, norm-aware BDI agents and virtual characters. Beyond the individual world view of virtual agents, this mechanism allows the analysis of social situations in which each virtual agent is situated, and thus leads them to 'rational' decision making in line with 'common sense' with assistance from social norms and social

reasoning: the former, social norms, may enlighten virtual agents to be capable of being conventionally polite in a society, whilst the latter may promise a better understanding of current social situations, so that it becomes a source of prediction about 'when' and 'what' to do in order to be interpersonally polite as perceived by other participants.

The main contribution of this chapter is a mechanism to cope with politeness in general rather than in a small specific aspect of daily activities. In reality, humans are likely to be engaged in many complicated social interactions so various kinds of polite activities are required corresponding to those situations. Likewise, virtual agents are also situated in similar circumstances due to major advances in virtual worlds, thus they need a higher level of autonomy to deal with a wider range of situations in which to be more polite.

With this aim in mind, the main focus of this chapter is to: (i) utilise **DNA**^{3}, a high level agent architecture combined with normative frameworks which capable of modelling, reasoning and deliberation about polite behaviours in social situations, and ***N-Jason*** which is able to deliberation on polite behaviour and agents own goals, (ii) demonstrate the interpretation of high level behaviour into sequence of physical atomic actions formed in animations in VEs, and (iii) show the evaluation of this approach.

6.1.2 Scenario

As mentioned in Section 6.1.1, Allen *et al.* [Allen et al., 2012] describe pioneering work on politeness in interactions between PCs and virtual characters. On the basis of an asymmetric relationship, virtual characters predict the next few movements of PCs using a Hidden Markov Model (HMM), and react politely such as changing velocity and path around PCs to avoid collisions. The result with regards to the number of collisions and completion time of navigation is remarkable, so the ease in navigation is achieved despite using low precision user input devices. However, offline learning is always required with training data representing spatial information and its related reactive behaviours in this system. This aspect gives rise to the issue on the lack of flexibility subject to changes in situations. When the envrionment where virtual characters reside is changed, the spatial information and associated behaviour are no longer valid.

In comparison to the work by Allen *et al.* [Allen et al., 2012], **DNA**^{3} has more

generality, making it applicable to circumstances without training data due to the nature of logic-based reasoning. In addition, we use autonomous and deliberative agents so richer behaviours may be presented with the combination of social reasoning and individual reasoning at any time, in any place. Thus, we show a simple illustrative example of how the concept of such politeness as proposed in [Allen et al., 2012] is achieved by **DNA**^{3} in Section 6.2. We here present a model, avoiding disturbing PC navigation [Allen et al., 2012], that is triggered when PCs are detected by virtual agents.

Then, we extend this individual level of politeness to the group behaviour level. In effect, polite behaviour in between groups of virtual characters requires more careful consideration due to social dynamics between members. In the previous scenario, politeness in navigation of individuals, the social relationship is relatively simple: there is only an asymmetry relationship between human player characters (PCs) and virtual characters (non-player characters, NPCs) so the polite behaviour is obvious, either avoiding disturbing PC navigation or not. However, social relationships in/between groups (composed of multiple members) and its underpinning implications are more complicated. Depending on social relationships amongst them, which potentially could bring about new social situations, the expression of politeness can be vary. According to the Social Importance Dynamics (SID) model [Mascarenhas et al., 2013], which is strongly based upon the status-power theory [Kemper, 2011], "*inter-personal Relation*", "*group membership and roles*" and "*task interdependence*" play a crucial role in the choice of behaviour of individual members within a group in practice. This implies that polite behaviour in groups could be affected by social situations embedding social status (e.g. interpersonal relationship, membership) as well as social power (e.g. roles or task representing responsibilities). It also reminds us that changes of situations, which give rise to the subsequent changes in social status and power, have impact on the determination of polite behaviour in that new situation.

As an example, let us assume a number of virtual characters are waiting for the guided tour in a museum. When there is a notice that the tour is starting from a tour guide, one of the polite behaviours of tourists is to become a member of the guided group. In this case, the tourists are likely to approach to the guide and keep a relatively close distance although they are not intimate enough to have a close inter-personal distance with each other. Afterwards, to follow the guide within a certain radius could be the other polite behaviour during the guided tour. If a collision between two groups is foreseen during the guided tour, usually the leader (e.g. a guide) of the group is

aware of avoiding disturbance of the other group as a polite behaviour. The guide group members are unlikely to give the same attention to the behaviour as they would when navigating as individuals. Instead, they just simply comply as the guide shows the way without being concerned about the collision[1].

Suddenly, there is a fire alarm in the museum. Several kinds of polite behaviour may be feasible as social norms to assist others: some guides may be able to point out the location of exits. Others may try to help group members to escape easily such as yielding a space, keeping order in crowds or coordinating people. Another possible behaviour just ignores everything suggested and follows its own goal (e.g. to escape as soon as possible without consideration of others). Depending on roles and relationship[2], these behaviour may be appropriate to rescue as social norms subject to the role in the group indicates[3].

As seen in the above example, the polite behaviour between groups depends on not only the social situation itself but also changes of social situations. In the rest part of the chapter, we would like to take into account this property in the case studies. To this end, Section 6.3 demonstrates the politeness in group behaviour described in the guided tour scenario in the previous paragraph. At the beginning, it is presented as the impact of social norms and their governance on the formation of groups subject to the human territoriality model [Scheflen and Ashcraft, 1976, Pedica and Högni Vilhjélmsson, 2010]. Then, we apply the same polite behaviour in navigation (the avoidance of disturbance of others presented in Section 6.2) on the group level. Afterwards, the change of politeness in group depending on the situation is shown in Section 6.4 with the example of an evacuation scenario in the emergency described above. In the end, after a short discussion on pros/cons and limitations of the experiments in Section 6.5, a summary is presented in Section 6.6.

The above two demonstrations are achieved by the process of: (i) specifying the polite behaviour as a form of social norms in the institutional model, (ii) social reasoning technique for the generation of social norms representing polite behaviour, (iii) deliberation on such norms, individual goals and sanctions in ***N-Jason*** agent, and (iv) realisation of the polite behaviour by atomic physical actions inside the virtual character.

[1]We call this a *guided tour scenario*.

[2]For example, one of the leader's roles might be to make sure the members' safety within a leader-member relationship

[3]We call this an *evacuation scenario*.

Note that in each case study section, we firstly describe the high level modelling and reasoning on politeness by the combination of insitution and ***N-Jason***. Then, we illustrate the practical implementation of polite behaviour in terms of actions and animations in VEs. It is followed by the results, which comprise experiment and evaluation, at the end of each section.

6.1.3 Objective

In Chapter 2 (particularly in Section 2.4), it is argued that the major drawbacks of state-of-the-art in norm-aware behaviour of virtual characters are in general:

1. A lack of capability in reasoning about norms subject to the changes of situations. Reminding the case that social norms are pre-defined in a single virtual characters mind. These norms are usually too domain specific thus no longer correct/adequate when the situation is changed.

2. A lack of norm autonomy in the deliberation of agents. In other words, virtual characters either cannot violate norms due to deterministic rules at design time or cannot conduct deliberation on norms, goals and sanctions with the consideration of both social and individual contextual knowledge.

In consequence, the main objective of this chapter is to verify the adequacy of **DNA**^{3} in the design and simulation of norm-awareness for virtual characters. In particular, we have in mind to show the characteristics of **DNA**^{3} with regards to

1. Flexibility in specifying and reasoning about norms themselves subject to changes of social situations (in response to the former drawback) and

2. Rationality, by means of norm-aware reasoning, to pursue deliberation over norms and agent's individual goals (in relation to the latter drawback).

Given the first case study with a scenario on politeness in navigation of individuals, we would like to address the advantage of the incorporation of social reasoning technique in the exhibition of polite behaviour, which is one of the benefits of a socio-cognitive perspective on decision making. Due to the nature of institutions, we may

expect individual virtual characters to have improved understanding of the social dimension, which in turn results in a better level of flexibility in social norms and associated behaviour.

With the second case study in the guided tour scenario, we aim to consider both flexibility and rationality. In this scenario, we can see a couple of situaion changes. One is 'gathering' as a group which affects the interpersonal distance of individuals in relation to social relationship between a guide and members. The other is 'navigation' between groups, which can be governed by the institution of the previous scenario. We show how these situations are managed by the additional flexibility in **DNA**^{3}. In addition, the choice of individual behaviour either driven by social norms or driven by agent's own goals is investigated. For example, assuming a norm for being polite in navigation is detached when a virtual character is a member of the guided tour group. Then the character should select a plan to be polite as an individual or as a member. By measuring the priority and deadline of each goal (e.g. deontic goal, ordinary goal), the rational choice can be made by norm-aware reasoning in ***N-Jason***.

The last case study with an evacuation scenario is an experiment covering both characteristics as well. Due to the fire alarm, it is obvious that there is a rapid change in the situation, which requires a different type of polite behaviour subject to the new environmental context. So this aspect can contribute to support the improved flexibility of **DNA**^{3}. However, the other emphasis of this case study lies in the latter: the enhancement in rationality. The fire alarm needs an immediate change of behaviour, whose priority is very high and deadline is very urgent. We show how norm-aware deliberation is able to deal with these properties in the determination of virtual character behaviour by the scheduling of intentions.

6.2 Politeness in Individual Navigation

In this section, we show a simple illustrative example of how the concept of politeness proposed in [Allen et al., 2012] is achieved by **DNA**^{3}, the combination of institutinal model, ***N-Jason***, and virtual characters. To do so, we present a collision avoidance model, which is triggered when PCs are detected by virtual agents.

This is a modelling of the theory of politeness composed of: (i) a formulation of a social norm representing a obligation, '*Avoid Collision*', and its related decision

making in high level agent architecture, for conventional politeness (described in Section 6.2.1), and (ii) a resolution of collision and a prediction of the future situation by the recognition of others' activities, which represents interpersonal politeness (illustrated in Section 6.2.2). Afterwards, the evaluation of this case study including experimental settings and its results is shown in Section 6.2.3.

6.2.1 High Level Modelling

Specifying Polite Behaviour in Navigation

As introduced in Chapter 4, we employ the declarative action language Inst*AL* introduced in Section 4.3.1 to construct the formal model of institution which specifies the social norms in the collision avoidance scenario. The *'Institution Specification'* in *'Normative Framework'* seen on the right of Figure 4-4 refers to this constructed institution.

We define three domain fluents denoting the friendship between a virtual character and a PC `friends(Ag, P)`, and two kinds of InterPersonal Distance (IPD) associating them: `lowIPD(Ag, P)` and `highIPD(Ag, P)` subject to the closeness of the friendship. When an intimate character is moving towards a virtual agent, then `lowIPD(Ag, P)` should be initiated to maintain a lower personal distance between them, otherwise `highIPD(Ag, P)` is initiated (see Listing 6.1)

Listing 6.1 : Domain Fluents

```
1 lowIPD(Ag, P) when friends(Ag, P);
2 highIPD(Ag, P) when not lowIPD(Ag, P);
```

As seen in Listing 6.2, an exogenous event is defined to indicate the physical event that a PC is detected, which then generates the corresponding institutional event:

Listing 6.2 : Events and Generation Rule

```
1 exogenous event detected(P);
2 inst event intDetected(P);
3 arrived(P) generates intArrived(P);
```

A set of consequence rules are also provided (see Listing 6.3) to specify the norms that should be initiated subject to the condition of the IPD: (i) when a high IPD is required, the agent is obliged and permitted to avoid collision with the player; (ii) when

a low IPD is required, the agent is obliged and permitted to greet the player.

Listing 6.3 : Consequence Rules

```
intArrived(P) initiates
    perm(avoidCollision(Ag, P)),
    obl(avoidCollision(Ag, P), deadline, violpoliteness(Ag))
    if highIPD(Agt, P);

intArrived(P) initiates
    perm(greet(Ag, P)),
    obl(greet(Ag, P), deadline, violpoliteness(Ag))
    if lowIPD(Agt, P);
```

Reasoning about Polite Behaviour using ASP

Once polite behaviours are specified using Inst*AL* , it can be then translated automatically into the a computational model in ASP, which sets the stage for the normative reasoning performed by the Answer Set Solver, *Clingo*. What follows is a fragment of ASP code for the initiation of the obligation norms translated from the Inst*AL* formal model (see Listing 6.4).

Listing 6.4 : ASP code translated from Inst*AL* model

```
initiated(obl(greet(Ag,P),
    deadline,violpoliteness(Ag)),I) :-
    occurred(intArrived(P),I),
    holdsat(live(ipd),I),
    holdsat(lowIPD(Ag,P),I),
    player(P), agent(Ag),instant(I).

initiated(obl(avoidCollision(Ag,P),
    deadline,violpoliteness(Ag)),I) :-
    occurred(intArrived(P),I),
    holdsat(live(ipd),I),
    holdsat(highIPD(Ag,P),I),
    player(P), agent(Ag),instant(I).
```

The process of realising social reasoning consists of: (i) translation of specifications from Inst*AL* into ASP code, (ii) generation of all possible answer sets using observed events, the rules of the specification, and the constraints on answer set generation, and (iii) querying the resulting answer sets for behavioural actions for the agents. Such a series of processes takes place in the *'Social Reasoning(ASP)'* component of the *'Normative Framework'* seen in Figure 4-4.

The query should result in the identification of 'correct' (somehow polite) behaviour as a form of social norms. A fragment of an answer set representing social norms is shown in Listing 6.5. It shows the normative consequences of a query following the observation of an external event that a player is detected:

Listing 6.5 : Answer Set representing Social Norms

```
Answer: 1
observed(detected(player1), t0).

holdsat(obl(avoidCollision(jason1,p1),
        deadline,
        violpoliteness(jason1)),t1)

holdsat(obl(greet(jason2,p1),
        deadline,
        violpoliteness(jason2)),t1)
SATISFIABLE
```

Provided with the friendship information that `jason2` is a friend of the player `p1`, whilst `jason1` is not, the normative consequences generated for agent `jason1` and `jason2` are shown as above. The agent `jason1` is obliged to perform action `avoidCollision` before the `deadline`, otherwise, a violation event `violpoliteness` is produced. For `jason2`, the obligation is to greet the player.

Decision Making with Norms in *N-Jason* Agents

As mentioned in Chapter 4, such normative consequences may become a part of belief or trigger sub-goals or goals presented as follows. At the same time, the ***N-Jason*** BDI agent (seen in the middle of Figure 4-4) then carries out the decision making, and sends a sequence of action plans to the virtual character to perform motor actions.

Listing 6.6 : An Example of *N-Jason* Agent Program

```
% Norms as (Sub-)Goals
@plan_avoid_collision[duration(30)]
+!avoidCollision(NPC, PC)
<- change(speed, low);
   change(orientation, degree(45));
   avoid_collision(true).

@plan_greeting[duration(10)]
+!greet(NPC, PC) : lowIPD & ~highIPD
<-  move(PC, close);
```

```
11      greeting(say).
12
13  @plan_greeting[duration(10)]
14  +!greeting : ~lowIPD & highIPD
15  <- greeting(bow).
16
17  % Norms as a Belief
18  @plan_be_polite[duration(30)]
19  +obl(ACT, D, P) : ACT = avoidCollision & ~lowIPD & highIPD
20  <- +!avoidCollision(_,_)[deadline(D), priority(P)].
```

Listing 6.6 is a fragment of ***N-Jason*** agent program for polite behaviour in individual navigation. Due to the run-time norm execution mechanism in ***N-Jason*** (described in Section 5.3.3), plans such as `@plan_avoid_collision` and `@plan_greeting` can be triggered when relevant obligations are adopted in agent mind. These plans inherit the deadline and priority that obligations have in its representation. Then ***N-Jason*** starts the norm-aware deliberation on (norm triggered) deontic goals and ordinary goals through intention scheduling with deadlines and priorities (shown in Section 5.3.4). In the following section, we introduce the actual realisation of those external actions.

6.2.2 Low Level Modelling

Prediction of Potential Collisions

The simple approach for the prediction of potential collisions is inspired by the social force model [Helbing and Molnar, 1995]. Both characters are modelled as cylinders, which are r_p in radius for PCs, and r_v for NPCs, respectively. Physical collisions occur when the distance between PCs and NPCs is less than $r_p + r_v$. Likewise, the scanning range is modelled as a cylinder, with a radius of R_s in order to verify the proximity between PC and NPC.

Once PC comes inside the scanning range R_s, then the NPC investigates and predicts using the following information: (i) vectors representing the direction of characters, determine whether collisions will occur or not (see Figure 6-1), and (ii) the intimacy determined by the inter-personal distance (IPD) [Hayduk, 1983], which allows a NPC to approach closer to a PC. The velocity is not considered since all agents have the same speed in this VE, but it would be straightforward to take this additional factor into account.

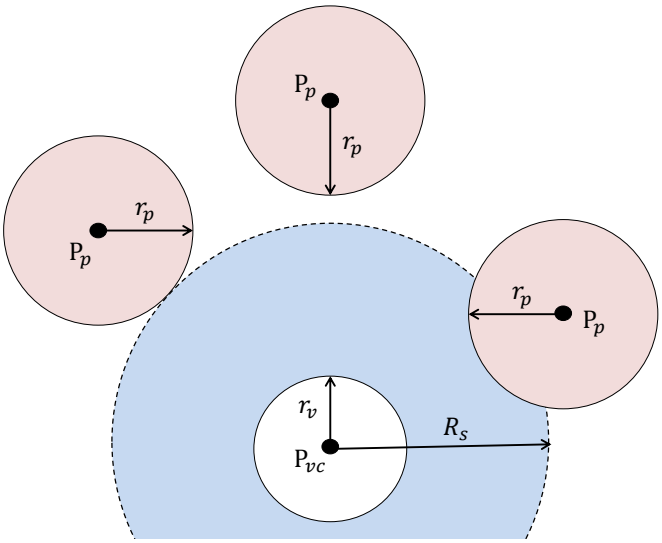

Figure 6-1: *Prediction of Potential Collisions*

Resolution of Physical Collisions

Once collision avoidance is decided upon, subject to the IPD between PC and NPC, the new position of a NPC is then updated by the same model proposed in [Allen et al., 2012] inspired by the Reciprocal Velocity Obstacle (RVO) algorithm [van den Berg et al., 2008].

RVO in principle takes into account a set of non-colliding velocities for NPC to follow, in order to avoid collisions amongst virtual characters. Based upon this, NPC attempts to reach its new location by setting a desirable velocity whenever the potential collision is predicted between PC and NPC. Given the current location of NPC at l_v and PC at l_p, the preferred velocity v_p for NPC to follow is simply

$$v_p = \frac{l_p - l_v}{\|l_p - l_v\|} \tag{6.1}$$

With the preferred velocity v_p, the new location $l_v{}'$ that avoids collision at l_v, the current position of NPC, is therefore determined by

$$l_v{}' = l_v + v_p((r_p + r_v) - \|l_p - l_v\|) \tag{6.2}$$

where v_p is a preferred velocity for NPC, l_p and l_v are the current location of PC and NPC, respectively, and r_p and r_v are the radii of PC and NPC, respectively. Equation 6.2

Figure 6-2: *Road Scenario*

implies that NPC is able to avoid the collision with PC by moving to the new location $l_v{}'$ from the current location l_v at a preferred velocity v_p when PC is moving towards the NPC's location l_v at a velocity v_p.

6.2.3 Evaluation

Experiments

We conduct a brief experiment chosen from [Allen et al., 2012] to evaluate the effectiveness of using polite behaviours for navigation in VEs. Two simple navigation scenarios are designed: navigation on the open road and in a doorway, seen in Figure 6-2 and 6-3, respectively. The main task of PCs is to reach the destination by passing through the group of avatars, which are moving in the opposite direction to the players.We assume that there may be moderate space between avatars which ensures players can walk between them without much effort.

In accordance with the scenario, the institutional model introduced in Section 6.2.1 is in charge of social reasoning in this experiment. The individual norm-aware cognitive reasoning is carried out by the agent program shown in Section 6.2.1.

The experiments are carried out in the *Second Life* [Linden Labs, 2014] virtual world. Players directly log in to the main server using its client program. These PCs are manipulated by an ordinary keyboard as usual for most human users. The virtual agents are created by libOMV [OpenMetaverse Organization, 2012], and are controlled by BDI reasoning agents. 10 virtual agents are used for a formation of a group in both scenarios. Each scenario is played out for 40 trials, split equally into 20 for polite

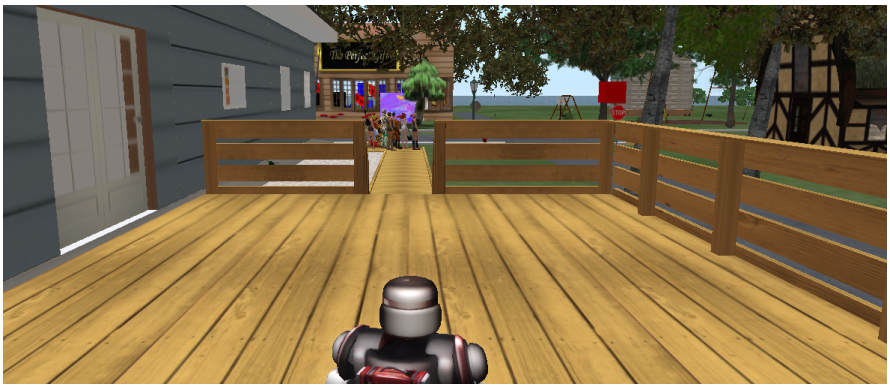

Figure 6-3: *Doorway Scenario*

agents and 20 for otherwise.

The collision detection is achieved programmatically by (i) capturing collision notifications from the *Second Life* server, and (ii) measuring distance at the same time, between PCs and virtual agents. Both are perceived by virtual agents internally. Note that we only take account of collisions between PCs and virtual agents in these experiments: PC-PC and NPC-NPC collisions are not considered.

Results

Overall statistics for collisions are shown in Table 6.1, where μ and σ are the mean and standard deviation, respectively.

Road			
	Type	μ	σ
Collisions	Polite	0.40	0.50
	Impolite	3.05	1.64

Doorway			
	Type	μ	σ
Collisions	Polite	0.55	0.60
	Impolite	2.10	0.97

Table 6.1: *Statistics of collisions*

These results suggest that polite agents provide quicker and easier navigation in crowded VEs. In both scenarios, we can say that passing through a group of polite agents is smoother and collision-free compared to that within a group of impolite agents.

Figure 6-4 and 6-5 also demonstrate the effectiveness of the politeness of virtual agents using social reasoning. In the first scenario, a high density group of agents provides a enough space for the PC to be able to pass through with little or no change

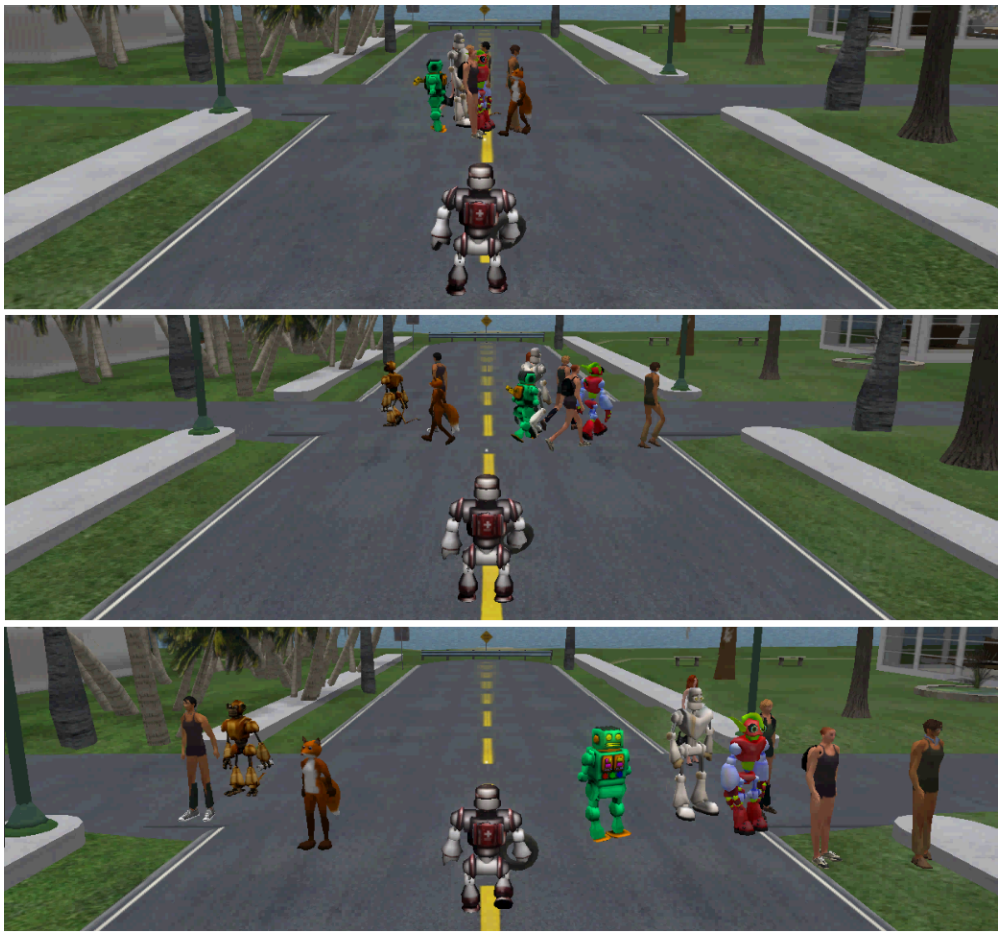

Figure 6-4: *Polite Behaviour on the Road*

of direction. Similarly, the player is rarely observed being obstructed by polite virtual characters despite the doorway being somewhat narrow and that many agents are still moving around in that space[4].

It may be appropriate to emphasise that all the virtual agents participating in this experiment are fully autonomous. There is no pre-training process or learning process involved, so each run will see different individual action sequences: the agents are not automata that do the same thing each time hence the necessity for multiple runs. In our model, agent autonomy is developed on the basis of temporal reasoning with sequence of event occurrences, which provides a degree of flexibility making it applicable for a range of circumstances. In addition, thanks to the use of autonomous and norm-aware

[4]The video clip is available via `http://people.bath.ac.uk/jl495/video/politeness.wmv`

Figure 6-5: *Polite Behaviour in Doorway*

N-Jason agents, more rich behaviours may be presented with the combination of social reasoning and individual reasoning at any time in any places.

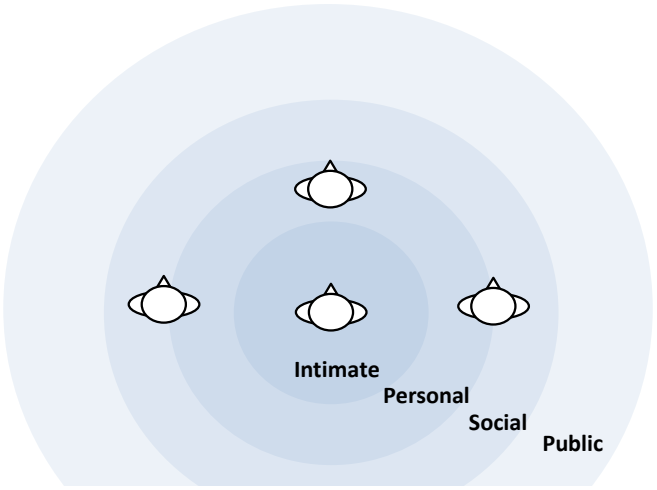

Figure 6-6: *The Concept Human Territories and Organisation*

6.3 Politeness in Group Navigation

As noted in Section 6.1.2, this section presents the second case study based upon the guided tour scenario. In principle, the main purpose of this section is to exhibit politeness between groups in the same way as it is shown between individuals (in particular between PCs and NPCs) in Section 6.2. To this end, we begin with the illustration of social dynamics between indviduals inside the group subject to Human Territoriality model [Scheflen and Ashcraft, 1976] (see Figure 6-6) introduced in Section 2.2.1. Then we describe polite behaviour between groups (i.e. avoiding disturbance of other group) whose underlying concept is described in Section 6.2.

The structure of this section is exactly the same as that of previous section. At the beginning (Section 6.3.1), we describe the high level model specifying social norms with respect to both the human territoriality model and politeness between groups, and its associated decision making in **DNA**^{3}. Afterwards, we present in Section 6.3.2 the low level modelling, which realises inter-group politeness driven by collision avoidance, as well as the Human Territoriality model [Scheflen and Ashcraft, 1976] in group formation. This is followed by the evaluation including experiment settings and results in Section 6.3.3.

6.3.1 High Level Modelling

Specifying Politeness in Group Formation

Same as the previous scenario in Section 6.2, we use Inst*AL* shown in Section 4.3.1 to construct the institution coordinating the behaviour on the group level. As discussed in Section 6.2.1, this constructed institution is an *'Institution Specification'* of the *'Normative Framework'* seen on the right of Figure 4-4. In here, we show a fragment of Inst*AL* code for the initiation of social norms (representing correct behaviour during the formation of a group) in response to exogenous events, generation rules and consequece rules.

We define exogenous events in order to indicate the physical events shown in Listing 6.7: (i) `see(Agent, Agent)` indicates the external events that a tour guide is seen by a NPC. (ii) `requestToJoin(Agent, Agent)` represents that the NPC sends a request to be a member of the tour guide's group. (iii) `agreeToJoin(Agent, Agent)` refers to the physical events that the tour guide agrees to the request. (iv) Then, `join(Agent, Agent)` can be detected as soon as the NPC joins as a member of the tour guide's group (physically, this means the NPC is now located in the member territory of the tour guide). (v) `follow` indicates the physical events that a tour guide is now moving. (vi) When the NPC has left the group, `escapeGroup(Agent, Agent)` is detected. Whenever exogenous events are detected, those events generate the corresponding institutional events.

Listing 6.7 : Exogenous Events and Generation Rules

```
1  exogenous event see(Agent, Agent);
2  see(Ag, AgLeader) generates
3      intSee(Ag, AgLeader);
4
5  exogenous event requestToJoin(Agent, Agent);
6  requestToJoin(Ag, AgLeader) generates
7      intRequestToJoin(Ag, AgLeader);
8
9  exogenous event agreeToJoin(Agent, Agent);
10 agreeToJoin(Ag, AgLeader) generates
11     intAgreeToJoin(Ag, AgLeader);
12
13 exogenous event join(Agent, Agent);
14 join(Ag, AgLeader) generates
15     intJoin(Ag, AgLeader);
16
```

```
17 exogenous event follow(Agent, Agent);
18 follow(Ag, AgLeader) generates
19     intFollow(Ag, AgLeader);
20
21 exogenous event escapeGroup(Agent, Agent);
22 escapeGroup(Ag, AgLeader) generates
23     intEscapeGroup(Ag, AgLeader);
```

A set of consequence rules are also provided to specify the norms that should be initiated subject to the current state as well as condition (see Listing 6.8). For example, when a NPC (denoted as `Ag`) finds a tour guide whose role is leader, the NPC is permitted to send a request to join, which in turn initiates the permission and obligation with `_DEADLINE_` and `_PRIORITY_`[5] to join the group of the tour guide (denoted as `AgLeader`) (line 1–18). As soon as the NPC joins the group, the role is changed to `member` (line 20–22). When the NPC has joined in the group, it is obliged to follow a tour guide within a given time (`_DEADLINE_`) and priority (`_PRIORITY_`).

Listing 6.8 : Consequence Rules

```
1  % if an agent sees a leader,
2  % then it's permitted to request join
3  intSee(Ag, AgLeader) initiates
4      perm(requestToJoin(Ag, AgLeader)),
5      perm(intRequestToJoin(Ag, AgLeader)),
6      pow(intRequestToJoin(Ag, AgLeader))
7      if roleOf(AgLeader, leader);
8
9  intRequestToJoin(Ag, AgLeader) initiates
10     perm(agreeToJoin(Ag, AgLeader)),
11     perm(intAgreeToJoin(Ag, AgLeader)),
12     pow(intAgreeToJoin(Ag, AgLeader));
13
14 intAgreeToJoin(Ag, AgLeader) initiates
15     perm(join(Ag, AgLeader)),
16     obl(join(Ag, AgLeader), _DEADLINE_, _PRIORITY_),
17     perm(intJoin(Ag, AgLeader)),
18     pow(intJoin(Ag, AgLeader));
19
20 % update the role of the agent Ag to be group member
21 intJoin(Ag, AgLeader) initiates
22     roleOf(Ag, member);
23
24 % agent Ag has to follow the leader
```

[5]For ease of reading, we use symbols representing a deadline and a priority instead of assigning real numbers.

```
% before the _DEADLINE_
intJoin(Ag, AgLeader) initiates
    obl(follow(Ag, AgLeader), _DEADLINE_, _PRIORITY_);

intJoin(Ag, AgLeader) initiates
    perm(follow(Ag, AgLeader)),
    perm(intFollow(Ag, AgLeader)),
    pow(intFollow(Ag, AgLeader));

intJoin(Ag, AgLeader) initiates
    perm(escapeGroup(Ag, AgLeader)),
    perm(intEscapeGroup(Ag, AgLeader)),
    pow(intEscapeGroup(Ag, AgLeader));

% terminates the role of agent Ag
% once it leaves the group
intEscapeGroup(Ag, AgLeader) terminates
    roleOf(Ag, member);;
```

Reasoning about Polite Behaviour using ASP

Same as the individual politeness model shown in Section 6.2.1, once polite behaviours in the group are specified using Inst*AL* , it can be then translated automatically into the a computational model in ASP, which sets the stage for the normative reasoning performed by the Answer Set Solver, *Clingo*. What follows is a fragment of ASP code for the initiation of the obligation norms translated from the Inst*AL* formal model (see Listing 6.9.

Listing 6.9 : ASP code translated from Inst*AL* model

```
initiated(obl(follow(Ag,AgLeader),_DEADLINE_,_PRIORITY_),group,I) :-
   occurred(intJoin(Ag,AgLeader),group,I),
   holdsat(live(group),group,I), inst(group),
   agent(Ag),
   agent(AgLeader),
   inst(group), instant(I).
```

The process of social reasoning in here is exactly same as the process shown in Section 6.3.1. Again, this social reasoning process takes place in the *'Social Reasoning(ASP)'* component of the *'Normative Framework'* in Figure 4-4. A fragment of an answer set representing social norms is shown in Listing 6.10. It shows the normative consequences of a query following the observation of an external event that a NPC is detected:

Listing 6.10 : Answer Set representing Social Norms

```
% Event Traces
observed(see(jason2, jason1), group, 0).
observed(requestToJoin(jason2, jason1), group, 1).
observed(agreeToJoin(jason2, jason1), group, 2).
observed(join(jason2, jason1), group, 3).

% Answer Set
Answer: 1
...
holdsat(obl(follow(jason2,jason1),_DEADLINE_,_PRIORITY_),group,4)
...

SATISFIABLE
```

Given the physical events updating status on the formation of the guided tour group between `jason2`, a NPC that wishes to be a member, and `jason1`, a tour guide, the normative consequences generated for virtual character `jason1` and `jason2` are as shown above. The agent `jason2` is obliged to perform action `follow` before the `_DEADLINE_` with `_PRIORITY_`, otherwise a violation occurs.

Decision Making with Norms in *N-Jason*

In this scenario, two types of ***N-Jason*** agents (seen at the middle of Figure 4-4) are used: the one is a leader agent acting as a tour guide. The other is a member agent acting as a member of guided tour. Basically, as shown in the first scenario in Section 6.2.1, normative consequences adopted by a ***N-Jason*** agent could trigger sub-goals or goals by run-time norm execution mechanism. Otherwise, the adopted norms may become a part of belief in the agent mind. Then, norm-aware deliberation begins to choose a plan to execute, taking into account deadlines and priorities.

Listing 6.11 is a fragment of a leader agent specification. When a certain NPC joins as a member of the leader agent's group, the leader agent updates a number of members and their names in its belief base (see `@plan_members_join`). Afterwards, the leader agent updates its territory mode and inter-personal distance with *pubcli* under the condition that the leader agent has the member(s). Otherwise, the territorial mode and its inter-personal distance become *social* (see `@plan_update_territory_social`). The duration of each plan is fixed as 30, and priority and deadline are set with the default values of infinity and zero, respectively.

Listing 6.11 : A Fragment of a Leader Agent Program

```
1  @plan_members_join[duration(30)]
2  +join(NAME) : not too_much(member) &
3     my_member(NAME) & not has_member(NAME)
4  <- .date(YY, MM, DD); .time(HH, NN, SS);
5     +add(YY, MM, DD, HH, NN, SS, member, NAME);
6     .print("Avatar", NAME, "is added.");
7     !update_territory_mode.
8
9  @plan_update_territory_public[duration(30)]
10 +!update_territory_mode : territory(T) &
11    .count(add(YY, MM, DD, HH, NN, SS, member, NAME), CNT) &
12    guide_limit(member, LIMIT) & CNT <= LIMIT & CNT > 0
13 <- +territory(public);
14    update_ipd(public).
15
16 @plan_update_territory_social[duration(30)]
17 +!update_territory_mode : territory(T) &
18   .count(add(YY, MM, DD, HH, NN, SS, member, NAME), CNT) &
19   CNT < 1
20 <- +territory(social);
21    update_ipd(social).
```

The leader agent in this scenario is governed by the institutinal model for polite behaviour in navigation of individuals shown in Section 6.2.1. When the leader (NPC in this case) detects the other leader (PC), then the plan for avoiding collisions is selected and thus executed with its own territory mode (either *public* or *social*) exactly the same as the agent in Section 6.2.1.

Listing 6.12 is a fragment of a member agent program. When the obligation `obl(follow(_,_),_DEADLINE_,_PRIORITY_)` is adopted, a plan `@plan_join_group` is triggered by the run-time norm execution mechanism in ***N-Jason*** in order to join a guided tour. Since the member agent is now a member of the leader, inter-personal distance between agent itself and a leader is zero which implies the member agent does not have to be governed by the institutional model for polite behaviour in navigation described in Section 6.2.1. Due to the characteristics of the run-time norm execution mechanism, deadline and priority of the plan `@plan_join_group_1` are same as those of the adopted obligation as explained in Section 5.3.3[6].

[6]See *Event-Reconsideration* in Section 5.3.3 for more details

Listing 6.12 : A fragment of a Member Agent Program

```
@plan_join_group[duration(30)]
+!follow(ROLE, NAME) : see(ROLE, NAME) & has_seen(ROLE)
<- .print("join_group");
   +territory(personal);
   +leader(NAME);
   update_speed(low);
   moveto(NAME);
   follow(NAME);
   update_ipd(personal).

@plan_escape_group_3[duration(30)]
+!escape_group(ROLE, NAME)[priority(5), deadline(60)]
: ROLE = guide & has_seen(guide)
<- -see(ROLE, NAME);
   -leader(NAME);
   update_speed(low);
   follow(none);
   update_ipd(social).
```

6.3.2 Low Level Modelling

Formation of Groups

As noted in Section 6.2.2, all virtual characters are modelled as cylinders inspired by social force model [Helbing and Molnar, 1995]. In the previous section, each character has a radius depending on the role, either PC or NPC. We in here extend this cylinderical model: each character has a radius depending on the role (either member or leader) and its associated territory mode. In this experiment, the radius of the leader character is either r_{pub} denoting *public territory* when be with members or r_{soc} representing *social territory* otherwise. In case of the group member agents, the radius is r_{per} denoting *personal territory* with a leader or r_{sco} as an individual. Figure 6-7 renders the idea on group formation based upon the concept of human territory seen in Figure 6-6. This formation is a means for the prediction of physical collision between groups in the following section.

Prediction of Potential Collisions between Groups

Given the formation shown in Figure 6-7, physical collision can be detected when the distance between groups is less than $2(r_{soc}+r_{per})$. The scanning range is also modelled as a cylinder with a radius of r_{pub}. We assume the scanning radius r_{pub} is always greater than $(r_{soc} + r_{per})$.

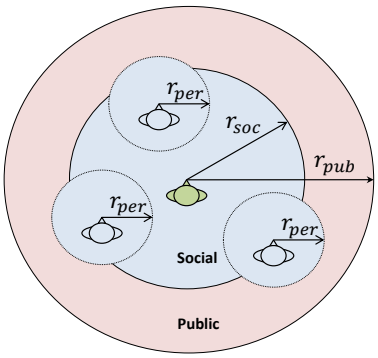

Figure 6-7: *Formation of Groups*

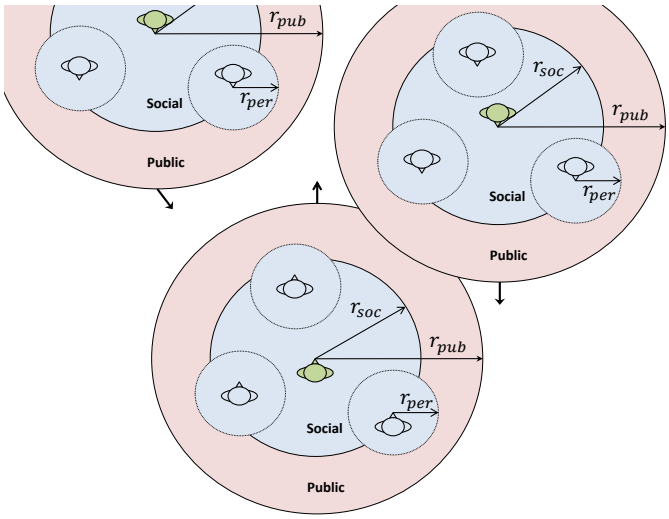

Figure 6-8: *Prediction of Potential Collisions between Groups*

The criterion for the prediction of collision detection shown in the right side of Figure 6-8 is exactly same as described in Section 6.2.2. Whenever the one group G comes inside the *public* territory of the other group G', then G' predicts the potential collision using vectors representing the direction of G, determines whether collisions will occur or not (see Figure 6-8). The velocity is not considered.

Resolution of Collisions between Groups

As introduced in Section 6.2.2, we use the same model poposed in [Allen et al., 2012], inspired by the Reciprocal Velocity Obstacle (RVO) algorithm [van den Berg et al., 2008], in this scenario when collision avoidance is decided (see Section 6.2.2 for more details).

Given the current location of the NPC tour guide (i.e. leader) at l_v and a PC tour guide at l_p, the preferred velocity v_p for the NPC tour guide to follow is simply

$$v_p = \frac{l_p - l_v}{\|l_p - l_v\|} \tag{6.3}$$

Then, the new location $l_v{}'$ is determined by

$$l_v{}' = l_v + v_p(2(r_{soc} + r_{per}) - \|l_p - l_v\|) \tag{6.4}$$

where v_p is a preferred velocity for NPC, l_p and l_v are the current location of PC and NPC tour guide, respectively, and r_{soc} and r_{per} are the radii of *social territory* and *personal territory*, respectively (see Figure 6-6). Equation 6.4 represents a guided tour group led by a NPC guide, so that it is able to avoid the collision with the other guided tour group led by a PC guide by moving to the new location $l_v{}'$ from the current location l_v at a preferred velocity v_p, when the PC's guided tour group is moving towards the NPC's location l_v at a velocity v_p.

6.3.3 Evaluation

Experiments

We conduct the same experiment chosen from [Allen et al., 2012] as we did in Section 6.2.3. In here, only one scenario of group navigation is considered: navigation in a museum seen in Figure 6-9. The main task of one group G led by the tour guide (i.e. leader, PC) is to reach the destination by passing through the other group G', which are moving in the opposite direction to the players.

In accordance with the scenario, two institutional models are involved in social reasoning in this experiment: the one is the model of '*Polite Behaviour in Navigation*' introduced in Section 6.2.1, and the other is '*Politeness in Group Formation*' model shown in 6.3.1. The individual norm-aware cognitive reasoning is carried out by two agents, a leader and a member, as described in Section 6.3.1.

The experimental settings are almost same as those in Section 6.2.3. One different factor is that 8 virtual agents are used for a formation of a group in this scenario. After an experiment on group formation, the experiment on politeness in group navigation is

Figure 6-9: *Museum Scenario*

played out for 40 trials, split equally into 20 for polite groups and 20 for otherwise. The collision detection is achieved programmatically by (i) capturing collision notifications from the *Second Life* server, and (ii) measuring distance at the same time, between groups. Both are perceived by virtual agents internally. The same as the previous setting, we only measure collisions between PCs and virtual agents and PC-PC and NPC-NPC collisions are not considered.

Results

Figure 6-10: *Polite Behaviour – Group Formation*

Figure 6-10 shows how individuals are gathering in order to form a group around a

tour guide (i.e. a leader). The male virtual character in a black t-shirt is a leader, and rest of them are members joining the guided tour[7].

The statistics for collisions are shown in Table 6.2, where μ and σ are the mean and standard deviation, respectively. The impression from this result is almost same as that in Section 6.2.3. This result with Figure 6-11 confirms that politeness in group behaviour also presents quicker and easier navigation experiences in VEs than with an ordinary group of agents.

Museum			
	Type	μ	σ
Collisions	Polite	0.50	0.59
	Impolite	5.95	1.53

Table 6.2: *Statistics of collisions Between Groups*

In this case study, same as the previous scenario, all the intelligent virtual agents are fully autonomous. There is no pre-training or learning undertaken, neither on the new environment (e.g. museum) nor on which plans to execute. The same applies to the institutional models. Both of them (agents and insitutions) are actually developed on the basis of logic-based reasoning with sequence of event occurences, which promises a degree of flexibility being applicable for a range of circumstances. For example, the institutional model governing the politeness in navigation successfully carries out the social reasoning thus regulates intelligent virtual agents behaviour regardless of environmeatal/situational changes in this experiment. Likewise, a norm-aware behaviour of virtual characters (avoiding disturbance of others in this case) is presented well under the same circumstances. Not only is the situation changed from individuals to groups navigation but also the environment changed from road, doorway to museum, the result, the pattern of statistics of collision avoidance is the same. These examples can be seen as an evidence that the logic-based approach to design both social and cognitive reasoning is able to improve the flexibility in presenting a degree of norm-aware behaviour of virtual characters at any time in any place.

[7]The video clip is available via `http://people.bath.ac.uk/jl495/video/group.wmv`

6.4 Politeness in Emergency Situation

This section presents the third case study, an evacuation scenario under the emergency situation as described in Section 6.1.2. When there is a fire alarm during a guided tour in the museum, virtual characters exhibit a different type of polite behaviour depending on their roles. Moreover they should act rationally with respect to urgency as well as importance, as polite behaviour requires. Given this case study, we expect to show both flexibility in social norms and rationality in decision making of **DNA**[3] as discussed in Section 6.1.3.

This section is organised as follows: In Section 6.4.1, the institutional model of the evacuation scenario and its associated agent programs are described. Then the low level modelling, an implementation side, is illustrated in Section 6.4.2. In the end, the evaluation including experimental settings and results is presented in Section 6.4.3.

6.4.1 High Level Modelling

Specifying Politeness under Emergency Situation

The institutional model in this section is relatively simpler than those in the previous sections (Section 6.2.1 and 6.3.1). Listing 6.13 shows a fragment of Inst*AL* code of the *Politeness under Emergency Situation* institution, which is *'Institution Specification'* in the *'Normative Framework'* on the right of Figure 4-4. Here, we firstly define the exogenous event `detectEmergency(Agent)` to indicate the physical event (fire alarm in this scenario) detected by a virtual character `Agent`. This exogenous event generates the corresponding institutional event `instOblEscape(Agent)` which in turn brings about the obligation `obl(exit_building(Agent), 1000, 100)` whose deadline and priority are 1000 and 100, respectively.

Listing 6.13 : Events and Generation/Consequence Rule

```
1 % Agent detects the occurrence of emergency (e.g. fire alarm)
2 exogenous event detectEmergency(Agent);
3 inst event instOblEscape(Agent);
4
5 % Generation Rule
6 detectEmergency(Agent) generates instOblEscape(Agent);
7
8 % Consequence Rule
```

```
9  instOblEscape(Agent) initiates
10     perm(exit_building(Agent)),
11     obl(exit_building(Agent), 1000, 100);
```

Reasoning about Polite Behaviour using ASP

Like the previous case studies, once polite behaviour is specified using Inst*AL* , it can be then translated automatically into the a computational model in ASP, which sets the stage for the normative reasoning performed by the Answer Set Solver, *Clingo*, taking place in the *'Social Reasoning(ASP)'* component of the *'Normative Framework'* in Figure 4-4. Listing 6.14 is a fragment of ASP code for the initiation of the obligation norms translated from the Inst*AL* formal model.

Listing 6.14 : ASP Code Translated from Inst*AL* model

```
1  initiated(obl(exit_building(Agent),1000,100),I) :-
2      occurred(iniOblEscape(Agent),I),
3      holdsat(live(queue),I),
4      agent(Agent),
5      instant(I).
```

A fragment of an answer set representing social norms is shown in Listing 6.15. It shows the normative consequences of a query following the observation of an external event that the fire alarm is detected:

Listing 6.15 : Answer Set representing Social Norms

```
1  % Event Traces
2  observed(createEvacuation,t0).
3  observed(detectEmergency(jason),t1).
4
5  % Answer Set
6  Answer: 1
7  ...
8  holdsat(obl(exit_building(jason),1000,100),t2)
9  ...
10
11 SATISFIABLE
```

Given the physical event, namely the detection of the fire alarm, the normative consequences generated for the agent *jason* are shown as above. The agent *jason* is now obliged to perform action `exit_building` before the `1000` with `100`, otherwise a violation occurs.

Decision Making with Norms in *N-Jason*

In this scenario, two types of ***N-Jason*** agents (seen at the middle of Figure 4-4) are used, a leader and a member, same as those in Section 6.2.1.

Listing 6.16 is a fragment of a leader agent in this scenario. When an obligation is detached, it triggers the goal with same deadline and priority the obligation imposes under run-time norm execution in ***N-Jason***. In response, an agent whose role is a tour guide (i.e. leader) has two choices. The one is direct engagement in the rescue of members as seen in the plan `@plan_escape_building_1`. When its member(s) are still left under its control, then the leader shouts out the escape message in order to inform to all mebers in the museum. If all members escape safely, then the leader agent itself now escapes the building. When there is no obligation detached but the leader agent perceives the fire alarm, the goal `+!exit_building` is triggered with deadline (50) and priority (20).

Listing 6.16 : A fragment of a Leader Agent Program in Evacuation

```
1  +fire_alarm : true
2  <- !exit_building(self)[deadline(50), priority(20)].
3
4  @plan_escape_building_1[duration(30)]
5  +!exit_building(AG)
6  : .count(add(YY, MM, DD, HH, NN, SS, member, NAME), CNT) & CNT > 0
7  <- shout(escape);
8     !exit_building(AG).
9
10 @plan_escape_building_2[duration(30)]
11 +!exit_building(AG)
12 : .count(add(YY, MM, DD, HH, NN, SS, member, NAME), CNT) & CNT < 1
13 <- update_ipd(none);
14    update_speed(fast);
15    moveto(exit).
```

Unlike the leader agent, the first option for members is to escape the building as soon as possible (see Listing 6.17). If there is still a leader with the group members, then the members may be able to follow the leader.

Listing 6.17 : A fragment of a Member Agent Program in Evacuation

```
1  @plan_escape_building_1[duration(30)]
2  +!exit_building(AG)
3  <- update_ipd(none);
4     update_speed(fast);
```

```
5 |    moveto(exit).
```

6.4.2 Low Level Modelling

As seen in Figure 6-12, there are three types of behaviour in the evacuation scenario. The most common behaviour (in particular for member agents) we can expect is to exit as soon as possible as shown on the right of Figure 6-12. The behaviour shown in both left and middle of Figure 6-12 is a polite behaviour of leader agents. They communicate to their group members to escape the building by shouting and waving hands.

6.4.3 Evaluation

Experiments

Continuing from the previous group navigation settings in Section 6.3.3, we conduct the experiment with a scenario of a evacuation model. During the guided tour, when the alarm is detected by one of virtual characters, the experiment starts.

In accordance with the scenario, two institutional models are involved in social reasoning in this experiment: the one is the model of '*Polite Behaviour in Navigation*' introduced in Section 6.2.1 in order to coordinate the group navigation, and the other is '*Politeness under Emergency Situation*' model shown in 6.4.1 for the evacuation scenario. The individual norm-aware cognitive reasoning is carried out by two agents, a leader and a member, as described in Section 6.3.1.

The experiment is conducted in the same place shown in Figure 6-9 with the same experimental settings described in Section 6.3.3. Here, we measure the response time of the virtual character when the obligation is detached. To see the enhancement of rationality in decision making of **DNA**[3]more accurately, we conduct the experiements with three configuration: (i) with ***N-Jason*** and the obligation which has the highest priority, (ii) with ***N-Jason*** and the obligation which has the lowest priority, and (iii) without ***N-Jason*** i.e. use a conventional ***Jason*** which is not capable of norm-aware reasoning.

The response time is a elapsed time between the time the obligation is adopted in the agent mind and the time the corresponding plan is executed[8]. The detection of time

[8]More precisely, we measure the time when the obligation is added as a goal triggering event in the event base (in `addEvent()`) and the time when the intention selected by *intention scheduling* is

stamp and calculation of elapsed time are achieved programmatically.

Results

The overall statistics for response time (in millisecond) are shown in Table 6.3, where μ and σ are the mean and standard deviation, respectively[9].

Museum			
	Agent Type	μ	σ
Response Time (ms)	***N-Jason*** (highest priority)	1.56	0.04
	N-Jason (lowest priority)	10.37	0.55
	Conventional ***Jason***	5.07	0.21

Table 6.3: *Statistics of Response Time*

Also Figure 6-13 demonstrates the result that how the evacuation is performed in the museum. When there is a fire alarm during a guided tour in the museum, every virtual character stops the current behaviour immediately. One group starts to escape the museum (the right side of the first image) but the other group (the left side of the first image). Once one of leaders starts and keep shouting, then the members of the leader are escaping the museum (the second and third image). In the end, every member successfully escapes the museum and waits for another notice (the fourth image). [10].

The result in Table 6.3 shows that the highest priority of a plan enables the most expeditious response. In constrast, the lowest priority causes the tardiest response. Compared to the case of using a conventional ***Jason*** which is not able to norm-aware deliberation, it is obvious that a degree of priority in norms and goals definitely affects the rational choice of behaviour. Assuming that a virtual character has a very important goal to achieve, and norms which the situaion requires participants to comply are less important to the virtual character. Then, it can be seen rational that the virtual character choose its individual goal instead of norms, and vice versa.

Given this result and its implication, the rationality in decision making on norms and goals can be improved by norm-aware deliberation in ***N-Jason***. In addition, this

excuted (in `reasoningCycle()`) just after the practical reasoning in ***N-Jason***

[9]Usually, three intentions are observed in the *intention scheduling*. The one is `+!exit_building(AG)` triggered by the obligation, and the other two is an intention triggered by the position update events from the virtual character (every 25ms) which is not seen in Listing 6.16.

[10]The video clip is available via `http://people.bath.ac.uk/jl495/video/group_escape.wmv`

norm-aware reasoning capability is able to contribute to the enhancement of norm-autonomy. According to the definition on autonomy by Castelfranchi [Castelfranchi, 1995], it could be an ability of an agent which can abandon something to do despite it is capable of doing that. Likewise, norm-autonomy is an ability to give up the compliance of the norm although an agent can comply the norm. ***N-Jason*** is able to do that by measuring the importance and imminence of goals and norms. In consequence, the compliance of norms can be abandoned thus violation is possible by norm-autonomy of ***N-Jason*** although it is capable of complying the norms.

6.5 Limitation

Notwithstanding the positive results above of the collision avoidance model for individuals and groups, and evacuation model that are achieved by the incorporation of social as well as norm-aware cognitive reasoning, there are some issues to discuss regarding the matter of taking measurements from experiments in *Second Life (SL)*.

The unusual definition of collision in *SL* weakens the reliability of the statistics of collision detection. In the *SL* server, collisions are only detected when one avatar's position is changed by being pushed by another. Only in this case is the collision actually detected and communicated to the client side. So, some cases which do not satisfy those condition may not be counted as collisions, even though they may be clearly observed by human eyes.

The client-server architecture is another factor interfering with the immediate detection of collisions. In the *SL* system, the metrics (such as position, orientation and velocity etc) of avatars come from the server to update client-local information, which might be used for monitoring the movement of others. However, packet loss, transmission delay and status up-date failure can all contribute to inaccuracy in movement detection. As a result, again, not all collisions may be counted.

Lastly, if an avatar keeps moving all the time, sometimes this avatar can not detect any event or the movement of other. This is serious because not all collisions can be reported, even if everything is witnessed.

For more precise and accurate measurement, it seems that manual counting of collisions using recorded video sequences [Allen et al., 2012] is the only reliable method. An alternative solution is to change to a different VE.

6.6 Summary

As we suggested at the outset, polite behaviour in virtual agents is promising for the efficiency and naturalism in navigation in dynamic virtual environments. To support this, we propose **DNA**^{3} combined with normative frameworks which is capable of modelling, reasoning and decision making about polite behaviour under social situations in conjuction with ***N-Jason*** which is able to provide norm-aware individual reasoning. Using this architecture, three case studies are explored. Firstly, avoiding disturbance of others' navigation [Allen et al., 2012], between PCs and virtual agents subject to inter-personal distance, is demonstrated as a simple example in a real-world 3D virtual environment. Then, the same navigation model between groups subject to Human Territoriality model [Scheflen and Ashcraft, 1976] is demonstrated in the same place. In both two studies, PCs are able to pass through a group of virtual agents, and navigate around them with relative ease. Afterwards, the evacuation model is presented at the end. Here, NPCs are able to show different types of polite behaviour depending on the different settings (e.g. roles) in the situations within rational response time the situations require.

Compared to the literature we cite, the framework developed as **DNA**^{3} has more generality and flexibility to be applied for all circumstances without pre-defined environmental data and its associated sets of behaviour. This is because it depends upon rule-based, temporal reasoning with a sequence of event occurrences, rather than a spatial reasoning systems which needs suitable training data. Also all forms of polite behaviours can be created subject to the theory of politeness thanks to the main functionalities in normative frameworks. In addition, we use norm autonomous and deliberative agents, by means of which more rational behaviours may be presented through the combination of social reasoning and individual reasoning at any time and in any place.

Figure 6-11: *Polite Behaviour – Groups Navigation in the Museum*

Figure 6-12: *Polite Behaviour – Types of Evacuation*

Figure 6-13: *Polite Behaviour – Evacuation*

Chapter 7

Conclusion

We now would like to recall Bandura's argument [Bandura, 2001].

> "*People do not live their life only in individual autonomy*."

In reality, people cannot always pursue what they really want to do: sometimes they give up their own desire depending on the social context. With the consideration of social factors in the context, individuals instead gladly comply with "*the way things are*" or "*the way things should be*" [Stein, 1993, Stein, 1997] when the behaviour is socially more worthy during their social activities. This implies that the determination of human behaviour is an outcome of the interplay between individual knowledge and social structures as proposed in socio-cognitive theory on human choices [Stein, 1993, Bandura, 2001, Akgün et al., 2003].

In this respect, it is not surprising that social intelligence substantially plays an important role in human choices. Norms have a potential to contribute to advances in this social intellience. As investigated in Section 4.2, the norms are socially constructed knowledge [Stein, 1993, Stein, 1997], which are established as a "*common sense*" through the social cognition process of actions and interactions of members in the society [Akgün et al., 2003]. This nature, socially constructed beliefs representing common sense, promises that norms not only offer a guidance about appropriate behaviour in the social situation, but also provide a sense of social expectations acting as an interpretation window of interactions in that situation. Thus, these characteristics of norms provide a good inspiration in the better comprehension of human behaviour, in particular, when the causality between what individuals really want to do and what they

can actually do is not explicilty seen in the social situation.

The domain of virtual characters research has accordingly given much attention to take take into account the benefits of norms in the design and simulation of virtual characters behaviour. However, as addressed in Chapters 1, 2 and 6, it has been challenging in the exhibition of naturalism in social behaviour of virtual characters due to: (i) a lack of capability in reasoning about norms subject to changes of situations, and (ii) a lack of norm-autonomy in individual reasoning. Within this context, the main objective in this thesis is to take advantage of such characteristics of norms in virtual agents' behaviour, possibly through the hybrid approach incorporating social structures (i.e. "*common sense*") and individual knowledge (i.e. "*private sense*") inspired by socio-cognitive theory [Stein, 1993, Bandura, 2001, Akgün et al., 2003].

Now we present the overall contributions and reflections in detail in Section 7.1. It is then followed by future work in Section 7.2.

7.1 Contribution

As discussed in Section 1.3, the primary contribution of this thesis is:

> "*The provision of Distributed Norm-Aware Agent Architecture (**DNA**^{3}) via the combination of social reasoning and cognitive reasong in order to engineer norm-aware behaviour in virtual characters*".

In theory, **DNA**^{3} pursues the faciliation of socio-cognitive perspective on the practical reasoning process of virtual characters by the combination of social (norms) reasoning on the social level and norm-aware cognitive reasoning on the individual level. In practice, **DNA**^{3} is a programming framework for the purpose of engineering norm-aware virtual characters, which is established by the integration of: (i) the Institution, also called a normative framework as a means to specifying and reasoning about norms, (ii) ***N-Jason***, a (BDI-type) cognitive agent carrying out the norm-aware practical reasoning and (iii) a virtual character in charge of perception and realisation of actions in virtual environments.

In Section 1.2, we defined that the challenges leading to research questions which have to be resolved in the design and simulation of norm-aware virtual characters. We show that how these research challenges are solved through **DNA**^{3} in the following.

1. The use of institutions in charge of social reasoning is a contribution of this thesis in response to Question 1[1] and 2[2] described in Chapter 1. Conceptually, the institution can be seen as a means of social cognition which constructs the external repositories of normative knowledge. The institution is in effect composed of a set of rules of which aim is the governance of individual agents in MAS. These rules describe consequences in response to knowledge or observations for reasoning about the current context. As a result, the institution brings about a set of situation-specific norms which can be a guidance representing appropriate/inappropriate behaviour in the form of obligations, permissions and prohibitions.

   Chapter 4 shows how this social reasoning is performed by the institutional model incorporating with individual virtual characters. Firstly, in order to construct the institutional model, we employ the declarative action language Inst*AL* introduced in Section 4.3.1 which specifies the social norms. Then, the Inst*AL* model can be translated automatically into the a computational model in ASP setting the stage for the normative reasoning performed by the Answer Set Solver, *Clingo*. Given this computational model, the social reasoning is accomplished through the generation of all possible answer sets using a sequence of event occurrences observed by multiple virtual agents. As a result, the institution brings about a new set of situationally appropriate behaviour depending upon the social context virtual characters encounter. Afterwards, the detachment of such a new set of norms (more precisely normative consequences of specific actions) to virtual characters takes place via run-time reasoning mechanism proposed in Section 4.4.

2. In consequence, such detached norms are adopted in virtual characters mind, which in turn involved in the practical reasoning process of individual agents. The other contribution of this thesis is the provision of ***N-Jason***, a norm-aware (BDI-type) cognitive agents proposed in Chapter 5 in response to Question 3[3]. The major role of ***N-Jason*** is to perform practical reasoning to select a plan to execute through the norm-aware deliberation on norms and individual goals. In principle, ***N-Jason*** offers a generic norm execution mechanism on top of norm-aware deliberation in order to contribute to run-time norm compliance. Thus, it

[1] "How to reason about norms subject to dynamic changes of situations in which virtual characters are involved"

[2] How to share the new set of norms (constructed by collectives) with virtual characters at run-time

[3] How to ensure norm-aware decision making in virtual characters' behaviour

is capable of the operationalisation of new and unknown norms not specified in the agent program at run-time by judging the executability of those norms. In the meantime, the selection of agent behaviour is achieved in the norm-aware reasoning process by *intention scheduling* with priorities and deadlines. This enables the evaluation of the relative importance and imminence between feasible plans triggered by both detached norms and goals, so that confirms the decision about which behaviour the agent would prefer.

3. Another contribution is the proposal of a middleware, the Bath-Sensor-Framework (BSF) introduced in Chapter 3 in response to Question 2 and 4. In the domain of intelligent virtual chatacters research, only a few virtual environments allow AI programming for their participants. This is mainly caused by heterogeneity such as differences in programming languages to develop the softwares and platforms (or OS) to run softwares as discussed in Section 3.1 and 3.4. The BSF enables the integration of these heterogeneous softwares as a middleware and its associated engineering methodology. With the pub/sub based communication on the basis of the standard network protocol (XMPP), BSF facilitates the integration of ***N-Jason***, virtual characters in the VE and the institution by supporting multiple programming languagues and physically distributed locations of softwares as a middleware shown in Chapter 3. In addition, the BSF provides communication channels in order to support the run-time sharing mechanism for observation and social information between ***N-Jason***, virtual characters and the institution with the concept of a (topic-based) node as seen in Chapter 4 (particularly in Section 4.4).

4. Taking advantage of **DNA**^{3} described above, three case studies, of which the main concept is politeness in virtual characters' behaviour, are conducted in Chapter 6. According to theory of politeness, polite behaviour should consider both conventional and interpersonal context in order to express appropriate politeness depending on the social situations participants encounter. In this sense, **DNA**^{3} is able to demonstrate politeness properly by understanding the situations which brings about situational-specific norms (driven by the institution) as well as considering the individual context and social context in agents mind (driven by norm-aware reasoning by ***N-Jason***). In here, polite behaviour of virtual characters is demonstrated, which can be differently presented according to the situations.

Firstly, avoiding disturbance of others' navigation [Allen et al., 2012], between PCs and virtual agents subject to inter-personal distance, is presented. Then, the same navigation model between groups is presented in a different place. Afterwards, the evacuation model is demonstrated. With the analysis of both statistics on the number of collisions in the first two experiments and statistics on the response time, we can conclude that the use of **DNA**^{3} contributes to advances not only in flexibility of reasoning about norms subject to changes of situations, but also rationality in decision making on norms and goals.

7.2 Future Works

Possible extensions to the work proposed in this thesis are manifold. We describe these future works in the context of the long-term goal, a provision of a social-aware intelligent (virtual) agent.

Facilitation of Multiple Institution

We firstly discuss the facilitation of multiple institions. The use of only a single insitution can be rather limited to specifying and reasoning about norms in the rich environments. In effect, the society where agents and humans are situated is not simple enough to be described by a single institution. Thus, the computational framework of the multiple institutions (MI) model is now considered to deal with richer situations in for virtual characters.

Actually, in Chapter 6, we use a preliminary form of MI shown in Figure 7-1. MI consists of two packages: the one is Institution Manager (InstManager afterwards) and the other is Web based Institution Service. InstManager is a gateway to use the MI connected by BSF between InstManager and a group of agents. The main role of InstManager is: (i) subcribe external events, (ii) update instances of institution in InstFactory (IF) and (iii) publish new normative states according to external events. Figure 7-2 shows the sequence of above operations in detail.

The concept of multiple institution enables various sets of institutional specifications which are reasoning about various kinds of norms such as social rules, cultural rules or laws. Perhaps, this rich set of institutions may be able to contribute to provide

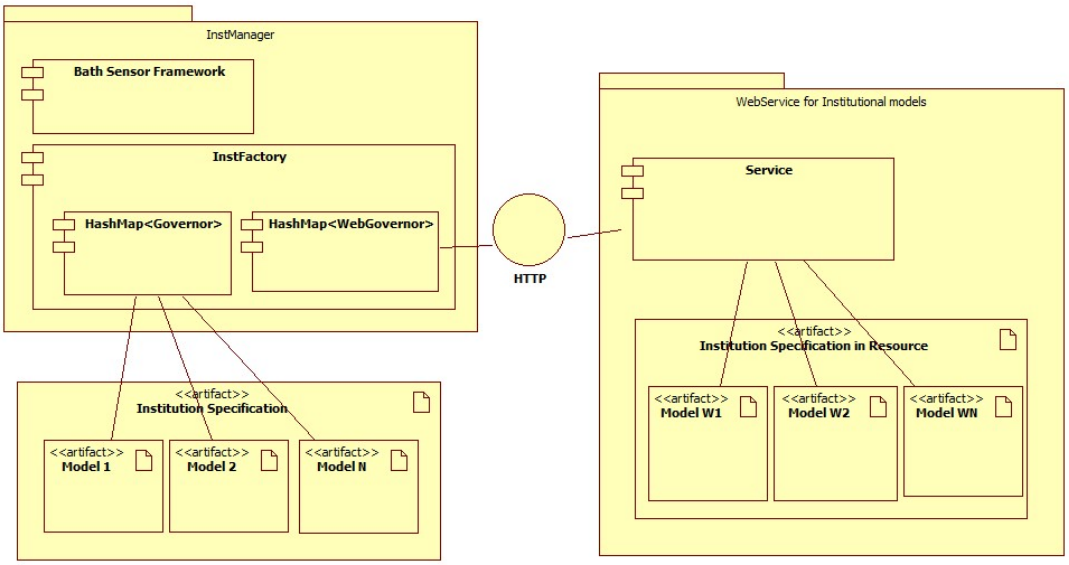

Figure 7-1: *A Preliminary Concept of Multiple Institutions*

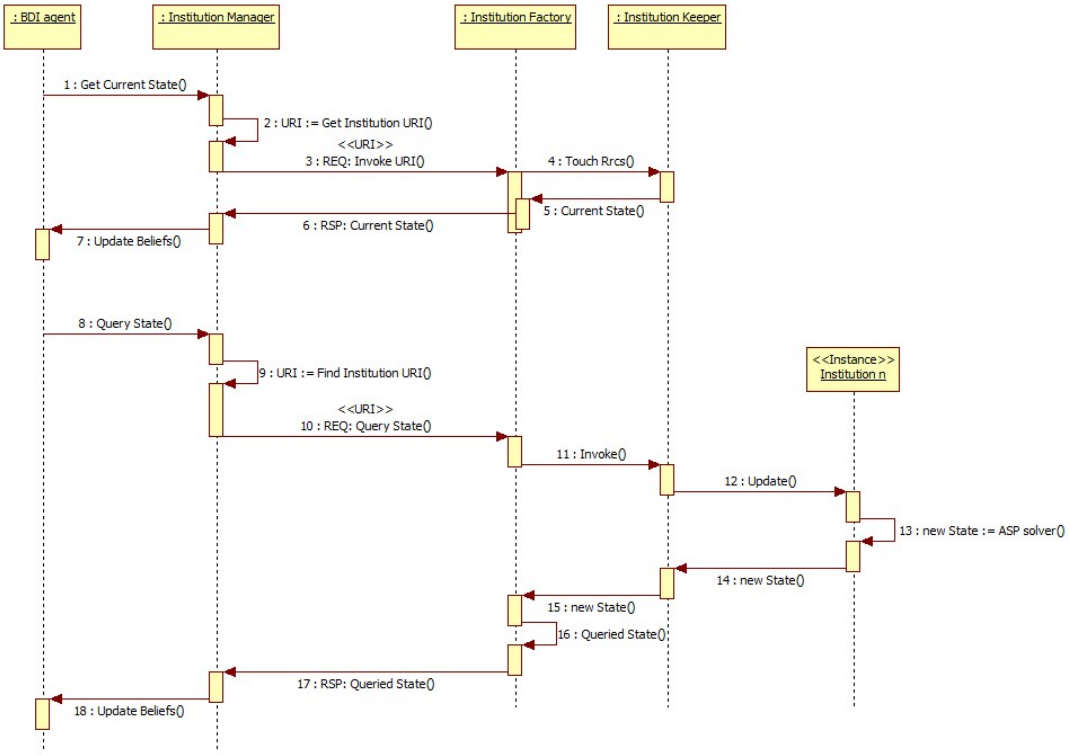

Figure 7-2: *A Sequence Diagram of Social Reasoning in Multiple Institutions*

more realistic social-awareness in virtual characters' behaviour by deeper and better understanding of situations. However, in this setting, conflicts between institutions should be considered. For this, the resolution of conflicts on the basis of inductive learning or conflict avoidance algorithms can resolve the tension here, as discussed clearly in [Li et al., 2013, Li, 2014].

Towards Social-Aware Cognitive Agents

Although ***N-Jason*** provides a level of rationality in deliberation on norms and individual goals, the more consideration is required to improve the norm-awareness.

First, norm-aware reasoning with deadlines and priorities requires a revision capability. Actually, this approach, norm-aware deliberation by intention scheduling with deadlines and priorities has a limitation due to pre-designed values (e.g. deadline, priority) in the agent programs. For more flexibility and robustness, these values should be re-assigned on the basis of adaptation to environments similar to the concept of dynamic assignment of a preference into plans proposed in [Padgham and Singh, 2013]. Depending on the agent types (e.g. BOID model [Broersen et al., 2002]) or the context of environments (e.g. social meetings, formal meetings or crisis), the values describing the internal characteristics of norms/goals should be re-evaluated in response to the adaptation process such as re-inforcement learning [Beheshti* et al., 2015]. The long term goal is to take into account reputations, emotions norms and individual goals with the identification of dynamics amongst them in a practical reasoning framework [Hoelz and Ralha, 2014] as human brains studies on social awareness which are investigated in sociology, psychology and economics.

To support the re-evaluation of values requires the detection of norms violation on the ***N-Jason*** interpreter level. For example, the priority of a certain norm should be increased when the norm violation happened in the agent mind, if the type of agent is social according to the BOID model [Broersen et al., 2002]. This type of violation, leading to the failure in execution of deontic plan that norms trigger, has an impact on the agent mental states. Under the normative settings, this violation can also occur in the process of norm adoption taking place in the belief update stage, and action selection process at the intention selection stage. Given the plan failure detection mechanism in BDI agents such as [Bordini and Hübner, 2010], we can formalise the violation representing the failure in execution of norms or when the normative goals are dropped

during scheduling, thus ensuring the more accurate changes in social awareness in agent mental states.

Another future works in which we are interested is the evaluation of the norm-aware agents. Like the analysis of semantics and properties shown in [Bordini and Hübner, 2010], [Alechina et al., 2012] and [Harland et al., 2014], operational semantics of norm aware agents can be fully formalised. In addition, the probabilistic modelling of norm-aware agent behaviour could be considered as it appears in [Winikoff and Cranefield, 2014, Vikhorev et al., 2011]. In conjunction with these theoretical evaluations, the experimental settings allow to measure the metrics such as response time of the agent subject to the adjusted values in norms and goals, or a number of plans in response to norms executed by the compliance mechanism as [Bordini et al., 2002] or [Waters et al., 2014] have done the comparative experimental evaluation of the agent reasoning capability.

Towards better Politeness in Virtual Characters

Given the combination of social-aware cognitive agents described in 7.2 and multiple institutions 7.2, we hope future research on politeness in virtual characters will enable more delicate polite behaviours through the extended social/cultural model, but also that politeness becomes usual amongst virtual characters, beyond the asymmetric relationship in players and virtual agents. We also hope these polite virtual characters will be able to interact human[4] which perhaps could be a contribution to the training purpose of serious games.

Politeness is inherently a qualitative judgement on behaviour, which would suggest the need for quite challenging – from a science and psychology perspective, as well as cost, time and repeatability – human-based studies. Consequently, we seek a lighter weight quantitative mechanism as a proxy for the qualitative study, for which factors such as number of changes of direction and the angular change involved in passing through the crowd might be suitable indicators of the quality of the avoidance mechanism. From an architectural perspective, multiple institutional models and related decision making mechanisms in BDI agents will be shown to provide a similar circumstance with real human society.

[4]The video clip is accessible via `http://cs.bath.ac.uk/~jl495/kinect.wmv`

Towards a Better Performance Evaluation of BSF

As we discussed in Chapter 3, the evaluation of performace for BSF is not simple. Plans for future work include a careful evaluation of performance issues, such as (i) the notion of component subscription profiles, (ii) monitoring of communication in live systems, so that out-of-profile situations can be detected, (iii) development of mitigations, such as throttling, replication and aggregation, (iv) experimentation with data handling policies, such as discarding and (finite) buffering, amongst others and (v) exploring the feasibility of applying corrective actions during execution, as well, of course, as the application of IVAs to more demanding scenarios.

Bibliography

[Adobbati et al., 2001] Adobbati, R., Marshall, A. N., Scholer, A., Tejada, S., Kaminka, G., Schaffer, S., and Sollitto, C. (2001). Gamebots: A 3D virtual world test-bed for multi-agent research. In *Proceedings of the Second International Workshop on Infrastructure for Agents, MAS, and Scalable MAS*, pages 47–52.

[Akgün et al., 2003] Akgün, A. E., Lynn, G. S., and Byrne, J. C. (2003). Organizational learning: A socio-cognitive framework. *Human Relations*, 56(7):839–868.

[Alechina et al., 2012] Alechina, N., Dastani, M., and Logan, B. (2012). Programming norm-aware agents. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems - Volume 2*, AAMAS '12, pages 1057–1064, Richland, SC. International Foundation for Autonomous Agents and Multiagent Systems.

[Allen et al., 2012] Allen, B., Magnenat-Thalmann, N., and Thalmann, D. (2012). Politeness improves interactivity in dense crowds. *Computer Animation and Virtual Worlds*, 23(6):569–578.

[Almajano et al., 2013] Almajano, P., Trescak, T., Esteva, M., Rodriguez, I., and Lopez Sanchez, M. (2013). v-mwater: An e-government application for water rights agreements. In Ossowski, S., editor, *Agreement Technologies*, volume 8 of *Law, Governance and Technology Series*, pages 583–595. Springer Netherlands.

[Anderson, 2005] Anderson, J. (2005). *Cognitive Psychology and Its Implications*. Worth Publishers.

[Anderson et al., 1997] Anderson, J., Matessa, M., and Lebiere, C. (1997). ACT-R: A theory of higher level cognition and its relation to visual attention. *Human Computer Interaction*, 12(4):439–462.

[Andrighetto et al., 2010] Andrighetto, G., Villatoro, D., and Conte, R. (2010). Norm internalization in artificial societies. *AI Commun.*, 23(4):325–339.

[Arcos et al., 2005] Arcos, J., Esteva, M., Noriega, P., and Sierra, C. (2005). C.: An integrated developing environment for electronic institutions. In *Agent Related Platforms, Frameworks, Systems, Applications, and Tools. Whitestein Book Series*. Springer Verlag.

[Arndt and Janney, 1985] Arndt, H. and Janney, R. (1985). Politeness revisited: Cross-modal supportive strategies. *IRAL: International Review of Applied Linguistics in Language Teaching*, 23(4):281–300.

[Baines and Padget, 2014] Baines, V. and Padget, J. (2014). On the benefit of collective norms for autonomous vehicles. In *Eighth International Workshop on Agents in Traffic and Transportation at AAMAS2014*.

[Balke et al., 2011] Balke, T., De Vos, M., Padget, J., and Traskas, D. (2011). On-line reasoning for institutionally-situated bdi agents. In *The 10th International Conference on Autonomous Agents and Multiagent Systems - Volume 3*, AAMAS '11, pages 1109–1110, Richland, SC. International Foundation for Autonomous Agents and Multiagent Systems.

[Bandura, 2001] Bandura, A. (2001). Social cognitive theory: An agentic perspective. *Annual Review of Psychology*, 52(1):1–26. PMID: 11148297.

[Bandura, 2005] Bandura, A. (2005). The evolution of social cognitive theory. *Great minds in management*, pages 9–35.

[Beheshti* et al., 2015] Beheshti*, R., Ali*, A. M., and Sukthankar, G. (2015). Cognitive social learners: an architecture for modeling normative behavior. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Austin, TX. (to appear).

[Bernstein and Vij, 2010a] Bernstein, D. and Vij, D. (2010a). Intercloud directory and exchange protocol detail using XMPP and RDF. In *Services (SERVICES-1), 2010 6th World Congress on*, pages 431–438.

[Bernstein and Vij, 2010b] Bernstein, D. and Vij, D. (2010b). Using XMPP as a transport in intercloud protocols. In *CloudComp, 2010 the 2nd International Conference on Cloud Computing*.

[Boella et al., 1999] Boella, G., Damiano, R., and Lesmo, L. (1999). Cooperating to the group's utility. In *Intelligent Agents VI*, pages 319–333. Springer Verlag.

[Boella and van der Torre, 2004] Boella, G. and van der Torre, L. (2004). Regulative and constitutive norms in normative multiagent systems. In *IN PROCS. OF KR04*, pages 255–265. AAAI Press.

[Boella et al., 2006] Boella, G., van der Torre, L., and Verhagen, H. (2006). Introduction to normative multiagent systems. *Computational Mathematical Organization Theory*, 12:71–79.

[Bogdanovych, 2007] Bogdanovych, A. (2007). *Virtual institutions*. PhD thesis, University of Technology Sydney.

[Bogdanovych et al., 2008a] Bogdanovych, A., Esteva, M., Simoff, S., Sierra, C., and Berger, H. (2008a). A methodology for developing multiagent systems as 3d electronic institutions. In Luck, M. and Padgham, L., editors, *Agent-Oriented Software Engineering VIII*, volume 4951 of *LNCS*, pages 103–117. Springer.

[Bogdanovych et al., 2012] Bogdanovych, A., Ijaz, K., and Simoff, S. (2012). The city of uruk: Teaching ancient history in a virtual world. In Nakano, Y., Neff, M., Paiva, A., and Walker, M., editors, *Intelligent Virtual Agents*, volume 7502 of *Lecture Notes in Computer Science*, pages 28–35. Springer Berlin Heidelberg.

[Bogdanovych et al., 2009] Bogdanovych, A., Rodriguez, J., Simoff, S., and Cohen, A. (2009). Virtual agents and 3d virtual worlds for preserving and simulating cultures. In *Intelligent Virtual Agents*, volume 5773 of *Lecture Notes in Computer Science*, pages 257–271. Springer Berlin Heidelberg.

[Bogdanovych et al., 2008b] Bogdanovych, A., Simoff, S., and Esteva, M. (2008b). Virtual institutions: Normative environments facilitating imitation learning in virtual agents. In Prendinger, H., Lester, J., and Ishizuka, M., editors, *Intelligent Virtual Agents*, volume 5208 of *Lecture Notes in Computer Science*, pages 456–464. Springer Berlin / Heidelberg.

[Bogdanovych et al., 2008c] Bogdanovych, A., Simoff, S., Esteva, M., and Debenham, J. (2008c). Teaching autonomous agents to move in a believable manner within virtual institutions. In Bramer, M., editor, *Artificial Intelligence in Theory and Practice*

II, volume 276 of *IFIP International Federation for Information Processing*, pages 55–64. Springer Boston.

[Boissier et al., 2013] Boissier, O., Bordini, R. H., Hübner, J. F., Ricci, A., and Santi, A. (2013). Multi-agent oriented programming with jacamo. *Sci. Comput. Program.*, 78(6):747–761.

[Boman, 1999] Boman, M. (1999). Norms in artificial decision making. *Artificial Intelligence and Law*, 7:17–35.

[Bordini et al., 2007] Bordini, R., Hübner, J., and Wooldridge, M. (2007). *Programming Multi-Agent Systems in AgentSpeak using Jason (Wiley Series in Agent Technology)*. John Wiley & Sons.

[Bordini et al., 2002] Bordini, R. H., Bazzan, A. L. C., de O. Jannone, R., Basso, D. M., Vicari, R. M., and Lesser, V. R. (2002). Agentspeak(xl): efficient intention selection in bdi agents via decision-theoretic task scheduling. In *Proceedings of the first international joint conference on Autonomous agents and multiagent systems: part 3*, AAMAS '02, pages 1294–1302, New York, NY, USA. ACM.

[Bordini and Hübner, 2010] Bordini, R. H. and Hübner, J. F. (2010). Semantics for the jason variant of agentspeak (plan failure and some internal actions). In *ECAI*, pages 635–640.

[Bourne et al., 1987] Bourne, L., Dominowski, R., and Loftus, E. (1987). Cognitive processes (2nd edn). *Applied Cognitive Psychology*, 1(1).

[Broersen et al., 2002] Broersen, J., Dastani, M., Hulstijn, J., and van der Torre, L. (2002). Goal generation in the boid architecture. *Cognitive Science Quarterly Journal*, 2 (3–4):428–447.

[Brom and Bryson, 2006] Brom, C. and Bryson, J. (2006). Action selection for intelligent systems. white paper 044-1, The European Network for the Advancement of Artificial Cognitive Systems.

[Brooks, 1991] Brooks, R. (1991). Intelligence without representation. *Artificial Intelligence*, 47(1-3):139–159.

[Brooks, 2001] Brooks, R. (2001). *How to build complete creatures rather than isolated cognitive simulators*. Architectures for Intelligence. Lawrence Erlbaum Associates, Mahwah.

[Brown and Levinson, 1987] Brown, P. and Levinson, S. (1987). *Politeness: Some universals in language usage*, volume 4. Cambridge University Press.

[Bryson, 2003] Bryson, J. J. (2003). Action selection and individuation in agent based modelling. In Sallach, D. L. and Macal, C., editors, *PROCEEDINGS OF AGENT 2003: CHALLENGES OF SOCIAL SIMULATION*, pages 317–330, Argonne, IL. Argonne National Laboratory.

[Castelfranchi, 1995] Castelfranchi, C. (1995). Guarantees for autonomy in cognitive agent architecture. In Wooldridge, M. and Jennings, N., editors, *Intelligent Agents*, volume 890 of *Lecture Notes in Computer Science*, pages 56–70. Springer Berlin Heidelberg.

[Charlton, 2000] Charlton, B. (2000). Evolution and the cognitive neuroscience of awareness, consciousness and language. *Cognition*, 50:7–15.

[Cliffe, 2007] Cliffe, O. (2007). *Specifying and Analysing Institutions in Multi-agent Systems Using Answer Set Programming*. PhD thesis, University of Bath.

[Cliffe et al., 2007] Cliffe, O., De Vos, M., and Padget, J. (2007). Specifying and reasoning about multiple institutions. In Noriega, P., Vázquez-Salceda, J., Boella, G., Boissier, O., Dignum, V., Fornara, N., and Matson, E., editors, *Coordination, Organizations, Institutions, and Norms in Agent Systems II*, volume 4386 of *Lecture Notes in Computer Science*, pages 67–85. Springer Berlin Heidelberg.

[Cliffe et al., 2009] Cliffe, O., De Vos, M., and Padget, J. (2009). Modelling normative frameworks using answer set programing. In Erdem, E., Lin, F., and Schaub, T., editors, *LPNMR*, volume 5753 of *Lecture Notes in Computer Science*, pages 548–553. Springer.

[Committee, 2012] Committee, O. A. M. Q. P. A. T. (2012). Advanced message queuing protocol 1.0. Technical report, OASIS. Available from `https://www.amqp.org/resources/download`, retrieved 20120416.

[Cook and Yanow, 1993] Cook, S. D. and Yanow, D. (1993). Culture and organizational learning. *Journal of management inquiry*, 2(4):373–390.

[Criado et al., 2010] Criado, N., Argente, E., and Botti, V. (2010). Normative deliberation in graded bdi agents. In Dix, J. and Witteveen, C., editors, *Multiagent System Technologies*, volume 6251 of *Lecture Notes in Computer Science*, pages 52–63. Springer Berlin Heidelberg.

[Dastani, 2008] Dastani, M. (2008). 2apl: a practical agent programming language. *Autonomous Agents and Multi-Agent Systems*, 16(3):214–248.

[Dastani et al., 2009] Dastani, M., Tinnemeier, N. A., and Meyer, J.-J. C. (2009). A programming language for normative multi-agent systems. *Multi-Agent Systems: Semantics and Dynamics of Organizational Models*, pages 397–417.

[Dautenhahn, 2007] Dautenhahn, K. (2007). Socially intelligent robots: dimensions of human–robot interaction. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 362(1480):679–704.

[De Vos et al., 2013] De Vos, M., Balke, T., and Satoh, K. (2013). Combining event- and state-based norms. In *Proceedings of the 2013 international conference on Autonomous agents and multi-agent systems*, AAMAS '13, pages 1157–1158, Richland, SC. International Foundation for Autonomous Agents and Multiagent Systems.

[Dignum, 1999] Dignum, F. (1999). Autonomous agents with norms. *Artificial Intelligence and Law*, 7:69–79. 10.1023/A:1008315530323.

[Dignum, 2004] Dignum, V. (2004). *A Model for Organizational Interaction*. PhD thesis, Utrecht University.

[Douglas, 1986] Douglas, M. (1986). *How institutions think*. Syracuse University Press.

[Duncan, 1979] Duncan, R. (1979). Organizational learning: Implications for organizational design. *Research in organizational behavior*, 1:75–123.

[Dybalova et al., 2013] Dybalova, D., Testerink, B., Dastani, M., and Logan, B. (2013). A framework for programming norm-aware multi-agent systems. In

Dignum, F. and Chopra, A., editors, *Proceedings of the 15th International Workshop on Coordination, Organisations, Institutions and Norms (COIN 2013)*.

[Easterby-Smith, 1997] Easterby-Smith, M. (1997). Disciplines of organizational learning: contributions and critiques. *Human relations*, 50(9):1085–1113.

[Esteva, 2003] Esteva, M. (2003). *Electronic Institutions: from specification to development*. PhD thesis, Technical University of Catalonia.

[Falcone and Castelfranchi, 2001] Falcone, R. and Castelfranchi, C. (2001). The human in the loop of a delegated agent: the theory of adjustable social autonomy. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 31(5):406 –418.

[Ferguson, 1992] Ferguson, I. A. (1992). Touring machines: Autonomous agents with attitudes. *Computer*, 25(5):51–55.

[Ferreira et al., 2012] Ferreira, N., Mascarenhas, S., Paiva, A., Dignum, F., Mc Breen, J., Degens, N., and Hofstede, G. (2012). Generating norm-related emotions in virtual agents. In Nakano, Y., Neff, M., Paiva, A., and Walker, M., editors, *Intelligent Virtual Agents*, volume 7502 of *Lecture Notes in Computer Science*, pages 97–104. Springer Berlin Heidelberg.

[Fikes et al., 1972] Fikes, R. E., Hart, P. E., and Nilsson, N. J. (1972). Learning and executing generalized robot plans. *Artificial Intelligence*, 3:251–288.

[Friedman et al., 2007] Friedman, D., Steed, A., and Slater, M. (2007). Spatial social behavior in second life. In *Intelligent Virtual Agents*, volume 4722 of *Lecture Notes in Computer Science*, pages 252–263. Springer Berlin Heidelberg.

[Gelfond and Lifschitz, 1991] Gelfond, M. and Lifschitz, V. (1991). Classical negation in logic programs and disjunctive databases. *New Generation Computing*, 9:365–386.

[Gemrot et al., 2009] Gemrot, J., Kadlec, R., Bída, M., Burkert, O., Píbil, R., Havlícek, J., Zemcák, L., Simlovic, J., Vansa, R., Stolba, M., Plch, T., and Brom, C. (2009). Pogamut 3 can assist developers in building ai (not only) for their videogame agents.

In Dignum, F., Bradshaw, J. M., Silverman, B. G., and van Doesburg, W. A., editors, *Agents for Games and Simulations*, volume 5920 of *Lecture Notes in Computer Science*, pages 1–15. Springer.

[Gibbs, 1981] Gibbs, J. P. (1981). *Norms, deviance, and social control: Conceptual matters*. Elsevier New York.

[Gioia and Sims, 1986] Gioia, D. A. and Sims, H. P. (1986). *The thinking organization*. San Francisco: Jossey-Bass.

[Granovetter, 1985] Granovetter, M. (1985). Economic action and social structure: the problem of embeddedness. *American journal of sociology*, pages 481–510.

[Group, a] Group, A.-R. R. ACT-R: Theory and architecture of cognition. `http://act-r.psy.cmu.edu/`.

[Group, b] Group, T. S. Soar project homepage. `http://sitemaker.umich.edu/soar`, retrieved 20120216.

[Harland et al., 2014] Harland, J., Morley, D., Thangarajah, J., and Yorke-Smith, N. (2014). An operational semantics for the goal life-cycle in bdi agents. *Autonomous Agents and Multi-Agent Systems*, 28(4):682–719.

[Harré, 1983] Harré, R. (1983). *Personal Being: A Theory for Individual Psychology*. Ways of being. Blackwell.

[Hayduk, 1983] Hayduk, L. (1983). Personal space: Where we now stand. *Psychological Bulletin*, 94(2):293–335.

[Helbing and Molnar, 1995] Helbing, D. and Molnar, P. (1995). Social force model for pedestrian dynamics. *Physical review E*, 51(5):4282–4286.

[Hilgard, 1980] Hilgard, E. R. (1980). The trilogy of mind: Cognition, affection, and conation. *Journal of the History of the Behavioral Sciences*, 16(2):107–117.

[Hindriks, 2008] Hindriks, K. (2008). Modules as policy-based intentions: Modular agent programming in goal. In Dastani, M., El Fallah Seghrouchni, A., Ricci, A., and Winikoff, M., editors, *Programming Multi-Agent Systems*, volume 4908 of *Lecture Notes in Computer Science*, pages 156–171. Springer Berlin Heidelberg.

[Hoelz and Ralha, 2014] Hoelz, B. and Ralha, C. (2014). Towards a cognitive meta-model for adaptive trust and reputation in open multi-agent systems. *Autonomous Agents and Multi-Agent Systems*, pages 1–32.

[Hogg and Jennings, 2000] Hogg, L. and Jennings, N. (2000). Variable sociability in agent-based decision making. In *Intelligent Agents VI. Agent Theories, Architectures, and Languages*, volume 1757 of *Lecture Notes in Computer Science*, pages 305–318. Springer Berlin / Heidelberg.

[Hollander and Wu, 2011] Hollander, C. D. and Wu, A. S. (2011). The current state of normative agent-based systems. *Journal of Artificial Societies and Social Simulation*, 14(2):6.

[Hubner et al., 2007] Hubner, J. F., Sichman, J. S., and Boissier, O. (2007). Developing organised multiagent systems using the moise+ model: programming issues at the system and agent levels. *Int. J. Agent-Oriented Softw. Eng.*, 1(3/4):370–395.

[Ignite Realtime, 0129a] Ignite Realtime (20130129a). Ignite realtime smack api homepage. Retrieved from `http://www.igniterealtime.org/projects/smack/`.

[Ignite Realtime, 0129b] Ignite Realtime (20130129b). The Openfire Project. `http://www.igniterealtime.org/projects/openfire/`.

[Jabber-Net,] Jabber-Net. The jabber.net project. `http://code.google.com/p/jabber-net`.

[Jive Software,] Jive Software. Openfire scalability. `http://www.igniterealtime.org/about/OpenfireScalability.pdf`, retrieved 20121109.

[Kahneman and Tversky, 1979] Kahneman, D. and Tversky, A. (1979). Prospect theory: An analysis of decision under risk. *Econometrica*, 47(2):pp. 263–292.

[Kemper, 2011] Kemper, T. D. (2011). *Status, power and ritual interaction: a relational reading of Durkheim, Goffman, and Collins*. Ashgate Publishing, Ltd.

[Kendon, 1990] Kendon, A. (1990). *Conducting Interaction: Patterns of Behavior in Focused Encounters*. Studies in Interactional Sociolinguistics. Cambridge University Press.

[Kim, 1998] Kim, D. H. (1998). The link between individual and organizational learning. *The strategic management of intellectual capital*, pages 41–62.

[Klimecki and Lassleben, 1998] Klimecki, R. and Lassleben, H. (1998). Modes of organizational learning indications from an empirical study. *Management Learning*, 29(4):405–430.

[Kumar et al., 2008] Kumar, S., Chhugani, J., Kim, C., Kim, D., Nguyen, A., Dubey, P., Bienia, C., and Kim, Y. (2008). Second life and the new generation of virtual worlds. *Computer*, 41(9):46–53.

[Larson and Christensen, 1993] Larson, J. R. and Christensen, C. (1993). Groups as problem-solving units: Toward a new meaning of social cognition. *British Journal of Social Psychology*, 32(1):5–30.

[Li, 2014] Li, T. (2014). *Normative Conflict Detection and Resolution in Cooperating Institutions*. PhD thesis, University of Bath.

[Li et al., 2013] Li, T., Balke, T., De Vos, M., Padget, J., and Satoh, K. (2013). A model-based approach to the automatic revision of secondary legislation. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Law*, ICAIL '13, pages 202–206, New York, NY, USA. ACM.

[Linden Labs, 2014] Linden Labs (2003-2014). Second life homepage. `http://www.secondlife.com`. Retrieved 20121211.

[Mascardi et al., 2005] Mascardi, V., Demergasso, D., and Ancona, D. (2005). Languages for programming bdi-style agents: an overview. In *WOA 2005 - Workshop From Objects to Agents*, pages 9–15.

[Mascarenhas et al., 2013] Mascarenhas, S., Prada, R., Paiva, A., and Hofstede, G. (2013). Social importance dynamics: A model for culturally-adaptive agents. In Aylett, R., Krenn, B., Pelachaud, C., and Shimodaira, H., editors, *Intelligent Virtual Agents*, volume 8108 of *Lecture Notes in Computer Science*, pages 325–338. Springer Berlin Heidelberg.

[Meneguzzi and Luck, 2009] Meneguzzi, F. and Luck, M. (2009). Norm-based behaviour modification in bdi agents. In *Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems - Volume 1*, AAMAS '09, pages

177–184, Richland, SC. International Foundation for Autonomous Agents and Multiagent Systems.

[Merritt et al., 2011] Merritt, T., Tan, K., Ong, C., Thomas, A., Chuah, T., and McGee, K. (2011). Are artificial team-mates scapegoats in computer games. In Hinds, P. J., Tang, J. C., Wang, J., Bardram, J. E., and Ducheneaut, N., editors, *CSCW*, pages 685–688. ACM.

[Messinger et al., 2009] Messinger, P. R., Stroulia, E., Lyons, K. A., Bone, M., Niu, R. H., Smirnov, K., and Perelgut, S. G. (2009). Virtual worlds - past, present, and future: New directions in social computing. *Decision Support Systems*, 47(3):204–228.

[Mitchell, 1973] Mitchell, J. (1973). *Networks, Norms and Institutions*. Mouton De Gruyter.

[Müller, 1996] Müller, J. (1996). The agent architecture InteRRaP. In *The Design of Intelligent Agents*, volume 1177 of *Lecture Notes in Computer Science*, pages 45–123. Springer Berlin / Heidelberg. 10.1007/BFb0017809.

[Neisser, 1976] Neisser, U. (1976). *Cognition and Reality: Principles and Implications of Cognitive Psychology*. A Series of Books in Psychology. W. H. Freeman.

[Nonaka and Takeuchi, 1995] Nonaka, I. and Takeuchi, H. (1995). *The knowledge-creating company: How Japanese companies create the dynamics of innovation*. Oxford university press.

[OpenMetaverse Organization, 2012] OpenMetaverse Organization (2012). libopenmetaverse developer wiki homepage. `http://lib.openmetaverse.org/wiki/`. Retrieved 20121217.

[Padgham and Singh, 2013] Padgham, L. and Singh, D. (2013). Situational preferences for bdi plans. In *Proceedings of the 2013 International Conference on Autonomous Agents and Multi-agent Systems*, AAMAS '13, pages 1013–1020, Richland, SC. International Foundation for Autonomous Agents and Multiagent Systems.

[Parsons, 1940] Parsons, T. (1940). The motivation of economic activities. *The Canadian Journal of Economics and Political Science/Revue canadienne d'Economique et de Science politique*, 6(2):187–202.

[Payr and Trappl, 2004] Payr, S. and Trappl, R. (2004). *Agent culture: human-agent interaction in a multicultural world*. CRC Press.

[Pedica and Högni Vilhjélmsson, 2010] Pedica, C. and Högni Vilhjélmsson, H. (2010). Spontaneous avatar behavior for human territoriality. *Applied Artificial Intelligence*, 24(6):575–593.

[Perreau de Pinninck et al., 2007] Perreau de Pinninck, A., Sierra, C., and Marco Schorlemmer, W. (2007). Distributed norm enforcement via ostracism. In Sichman, J. S., Padget, J. A., Ossowski, S., and Noriega, P., editors, *COIN*, volume 4870 of *Lecture Notes in Computer Science*, pages 301–315. Springer.

[Pokahr et al., 2005] Pokahr, A., Braubach, L., and Lamersdorf, W. (2005). Jadex: A bdi reasoning engine. In Bordini, R., Dastani, M., Dix, J., and El Fallah Seghrouchni, A., editors, *Multi-Agent Programming*, volume 15 of *Multiagent Systems, Artificial Societies, and Simulated Organizations*, pages 149–174. Springer US.

[Ranathunga, 2013] Ranathunga, S. (2013). *Improving Awareness of Intelligent Virtual Agents*. PhD thesis, The University of Otago.

[Ranathunga et al., 2011] Ranathunga, S., Cranefield, S., and Purvis, M. (2011). Interfacing a cognitive agent platform with a virtual world: a case study using second life. In *The 10th International Conference on Autonomous Agents and Multiagent Systems - Volume 3*, AAMAS '11, pages 1181–1182, Richland, SC. International Foundation for Autonomous Agents and Multiagent Systems.

[Ranathunga et al., 2012] Ranathunga, S., Cranefield, S., and Purvis, M. (2012). Identifying events taking place in Second Life virtual environments. *Applied Artificial Intelligence*, 26(1–2):137–181.

[Rao, 1996] Rao, A. (1996). Agentspeak(l): Bdi agents speak out in a logical computable language. In Van de Velde, W. and Perram, J., editors, *Agents Breaking Away*, volume 1038 of *Lecture Notes in Computer Science*, pages 42–55. Springer Berlin Heidelberg.

[Rao and Georgeff, 1995] Rao, A. S. and Georgeff, M. P. (1995). BDI agents: from theory to practice. In *Proceedings of the First Intl. Conference on Multiagent Systems*, San Francisco.

[Reed, 2012] Reed, S. (2012). *Cognition: Theories and applications*. CENGAGE learning.

[Ricci et al., 2011] Ricci, A., Piunti, M., and Viroli, M. (2011). Environment programming in MAS : An artifact-based perspective. *Autonomous Agents and MultiAgent Systems*, 23(2):158–192.

[Rosenfeld et al., 1984] Rosenfeld, H. M., Breck, B. E., Smith, S. H., and Kehoe, S. (1984). Intimacy-mediators of the proximity-gaze compensation effect: Movement, conversational role, acquaintance, and gender. *Journal of Nonverbal Behavior*, 8:235–249.

[Roy, 2007] Roy, P. V. (2007). Self management and the future of software design. *Electr. Notes Theor. Comput. Sci.*, 182:201–217.

[Russell and Norvig, 2002] Russell, S. J. and Norvig, P. (2002). *Artificial Intelligence: A Modern Approach, 2nd Ed.*, volume 2. Prentice Hall, Englewood Cliffs, NJ.

[Samperi et al., 2012] Samperi, K., Beale, R., and Hawes, N. (2012). Please keep off the grass: Individual norms in virtual worlds. In *Proceedings of the 26th Annual BCS Interaction Specialist Group Conference on People and Computers*, BCS-HCI '12, pages 375–380, Swinton, UK, UK. British Computer Society.

[Savarimuthu et al., 2008] Savarimuthu, B. T. R., Purvis, M., and Purvis, M. (2008). Social norm emergence in virtual agent societies. In *Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems - Volume 3*, AAMAS '08, pages 1521–1524, Richland, SC. International Foundation for Autonomous Agents and Multiagent Systems.

[Scheflen and Ashcraft, 1976] Scheflen, A. and Ashcraft, N. (1976). *Human Territories: How We Behave in Space-Time*. A Spectrum book ; S-375. Prentice-Hall.

[Schotter, 2008] Schotter, A. (2008). The economic theory of social institutions. *Cambridge Books*.

[Searle, 1995] Searle, J. (1995). *The construction of social reality*. New York: The Free Press.

[Si et al., 2006] Si, M., Marsella, S., and Pynadath, D. (2006). Thespian: Modeling socially normative behavior in a decision-theoretic framework. In *Intelligent Virtual Agents*, pages 369–382. Springer.

[Simmel and Wolff, 1950] Simmel, G. and Wolff, K. (1950). *The Sociology of Georg Simmel*. Free Press paperback. Free Press.

[Simon, 1990] Simon, H. A. (1990). Invariants of human behavior. *Annual Review of Psychology*, 41(1):1–20. PMID: 18331187.

[Smallwood and Sondik, 1973] Smallwood, R. D. and Sondik, E. J. (1973). The optimal control of partially observable markov processes over a finite horizon. *Operations Research*, 21(5):1071–1088.

[Stein, 1993] Stein, J. (1993). *Strategy Formation and Managerial Agency: a socio-cognitive perspective*. Doctoral thesis, Stockholm School of Economics, Management and Organization.

[Stein, 1997] Stein, J. (1997). How institutions learn: a socio-cognitive perspective. *Journal of Economic Issues*, pages 729–740.

[Stout et al., 2009] Stout, L., Murphy, M. A., and Goasguen, S. (2009). Kestrel: an XMPP-based framework for many task computing applications. In *Proceedings of the 2nd Workshop on Many-Task Computing on Grids and Supercomputers*, MTAGS '09, pages 11:1–11:6, New York, NY, USA. ACM.

[Therborn, 2002] Therborn, G. (2002). Back to norms! on the scope and dynamics of norms and normative action. *Current Sociology*, 50(6):863–880.

[Trescak et al., 2011] Trescak, T., Esteva, M., and Rodriguez, I. (2011). Vixee an innovative communication infrastructure for virtual institutions. In *The 10th International Conference on Autonomous Agents and Multiagent Systems - Volume 3*, AAMAS '11, pages 1131–1132, Richland, SC. International Foundation for Autonomous Agents and Multiagent Systems.

[van den Berg et al., 2008] van den Berg, J., Lin, M., and Manocha, D. (2008). Reciprocal velocity obstacles for real-time multi-agent navigation. In *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, pages 1928–1935.

[van Oijen et al., 2012] van Oijen, J., Vanhée, L., and Dignum, F. (2012). CIGA: A middleware for intelligent agents in virtual environments. In Beer, M., Brom, C., Dignum, F., and Soo, V.-W., editors, *Agents for Educational Games and Simulations*, volume 7471 of *Lecture Notes in Computer Science*, pages 22–37. Springer.

[van Riemsdijk et al., 2013] van Riemsdijk, M. B., Dennis, L. A., Fisher, M., and Hindriks, K. V. (2013). Agent reasoning for norm compliance: a semantic approach. In *Proceedings of the 2013 international conference on Autonomous agents and multi-agent systems*, AAMAS '13, pages 499–506, Richland, SC. International Foundation for Autonomous Agents and Multiagent Systems.

[Veksler, 2009] Veksler, V. (2009). Second-life as a simulation environment: Rich, high-fidelity world, minus the hassles. In *Proceedings of the 9th International Conference of Cognitive Modeling*.

[Verhagen and Boman, 1999] Verhagen, H. and Boman, M. (1999). Norms can replace plans. In *IJCAI-99 Workshop on Adjustable, Autonomous Systems*.

[Vikhorev et al., 2011] Vikhorev, K., Alechina, N., and Logan, B. (2011). Agent programming with priorities and deadlines. In *The 10th International Conference on Autonomous Agents and Multiagent Systems - Volume 1*, AAMAS '11, pages 397–404, Richland, SC. International Foundation for Autonomous Agents and Multiagent Systems.

[Wagener et al., 2009] Wagener, J., Spjuth, O., Willighagen, E., and Wikberg, J. (2009). XMPP for cloud computing in bioinformatics supporting discovery and invocation of asynchronous web services. *BMC Bioinformatics*, 10(1):279.

[Wallace, 1983] Wallace, W. (1983). *Principles of scientific sociology*. New York: Aldine.

[Walters et al., 2008] Walters, M. L., Syrdal, D., Dautenhahn, K., teBoekhorst, R., and Koay, K. L. (2008). Avoiding the uncanny valley: robot appearance, personality and consistency of behavior in an attention-seeking home scenario for a robot companion. *Autonomous Robots*, 24(2):159–178.

[Waters et al., 2014] Waters, M., Padgham, L., and Sardina, S. (2014). Evaluating coverage based intention selection. In *Proceedings of the 2014 International Con-*

ference on Autonomous Agents and Multi-agent Systems, AAMAS '14, pages 957–964, Richland, SC. International Foundation for Autonomous Agents and Multiagent Systems.

[Weber, 2009] Weber, M. (2009). *The theory of social and economic organization*. Simon and Schuster. reprinted 2009.

[Weick, 1979] Weick, K. E. (1979). The social psychology of organizing (topics in social psychology series).

[Weick, 1988] Weick, K. E. (1988). Enacted sensemaking in crisis situations[1]. *Journal of Management Studies*, 25(4):305–317.

[Weiner et al., 1983] Weiner, B., Graham, S., Taylor, S. E., and Meyer, W.-U. (1983). Social cognition in the classroom. *Educational Psychologist*, 18(2):109–124.

[Winikoff, 2005] Winikoff, M. (2005). Jack intelligent agents: An industrial strength platform. In Bordini, R., Dastani, M., Dix, J., and El Fallah Seghrouchni, A., editors, *Multi-Agent Programming*, volume 15 of *Multiagent Systems, Artificial Societies, and Simulated Organizations*, pages 175–193. Springer US.

[Winikoff and Cranefield, 2014] Winikoff, M. and Cranefield, S. (2014). On the testability of bdi agent systems. *Journal of Artificial Intelligence Research*, 51:71–131.

[Wooldridge, 2009] Wooldridge, M. (2009). *An Introduction to MultiAgent Systems (Second Edition)*. Wiley.

[Wyer Jr and Srull, 1984] Wyer Jr, R. S. and Srull, T. K. (1984). *Handbook of Social Cognition: Volume 1: Basic Processes Volume 2: Applications*. Psychology Press.

[XEP-301: In-Band Real Time Text, 2012] XEP-301: In-Band Real Time Text (2012). Xep-301: In-band real time text. Technical report, XMPP Standards Foundation. Available from `http://xmpp.org/extensions/xep-0301.pdf`, retrieved 20120416.

[XMPP Standards Foundation, 0129a] XMPP Standards Foundation (20130129a). Extensible messaging and presence protocol(XMPP): Core, and related other RFCs. `http://xmpp.org/rfcs/rfc3920.html`.

[XMPP Standards Foundation, 0129b] XMPP Standards Foundation (20130129b). Xep-0060: Publish-subscribe. Retrieved from http://www.xmpp.org/extensions/xep-0060.html,.

[Yee et al., 2007] Yee, N., Bailenson, J. N., Urbanek, M., Chang, F., and Merget, D. (2007). The unbearable likeness of being digital: The persistence of nonverbal social norms in online virtual environments. *Cyberpsy., Behavior, and Soc. Networking*, 10(1):115–121.